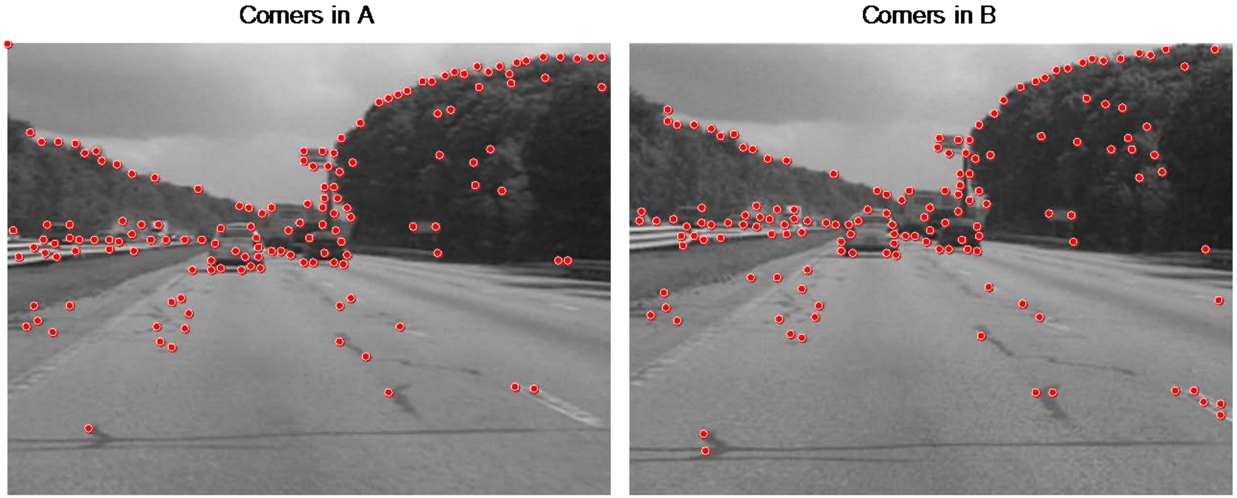
CS 4495 Computer Vision

Features 1 – Harris and other corners

Aaron Bobick
School of Interactive Computing

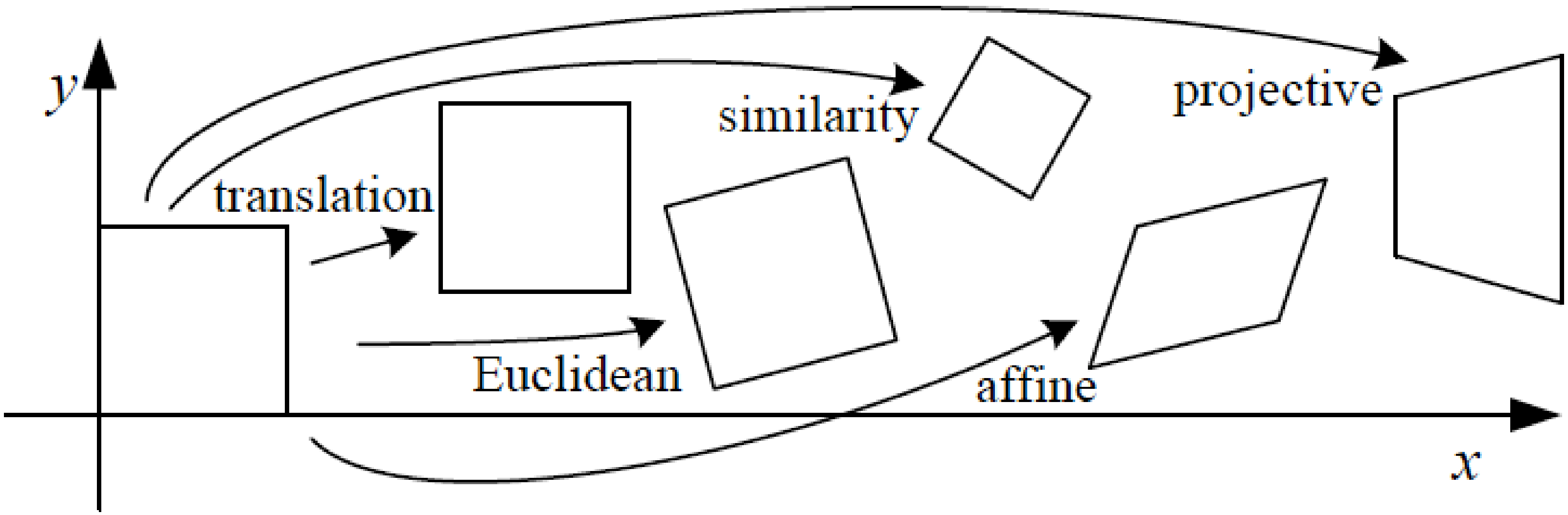


Administrivia

- PS 4: Out and not changed. Due Sunday Oct 12th, 11:55pm
  - It is application of the last few lectures. Mostly straight forward Matlab but if you’re linear algebra is rusty it can take a while to figure out. Question 2 takes some doing to understand.
    - You have been warned…
  - It is cool
    - You have been warned…
- Today: Start on *features.*
  - Forsyth and Ponce: 5.3-5.4
  - Szeliski also covers this well – Section 4 – 4.1.1
  - These next 3 lectures will provide detail for Project 4.

The basic image point matching problem

- Suppose I have two images related by some transformation. Or have two images of the same object in different positions.
- How to find the transformation of image 1 that would align it with image 2?



We want ***Local***(1) ***Features***(2)

- Goal: Find points in an image that can be:
  - Found in other images
  - Found precisely – well localized
  - Found reliably – well matched

- Why?
  - Want to compute a fundamental matrix to recover geometry
  - Robotics/Vision: See how a bunch of points move from one frame to another. Allows computation of how camera moved -> depth -> moving objects
  - Build a panorama…

Suppose you want to build a panorama

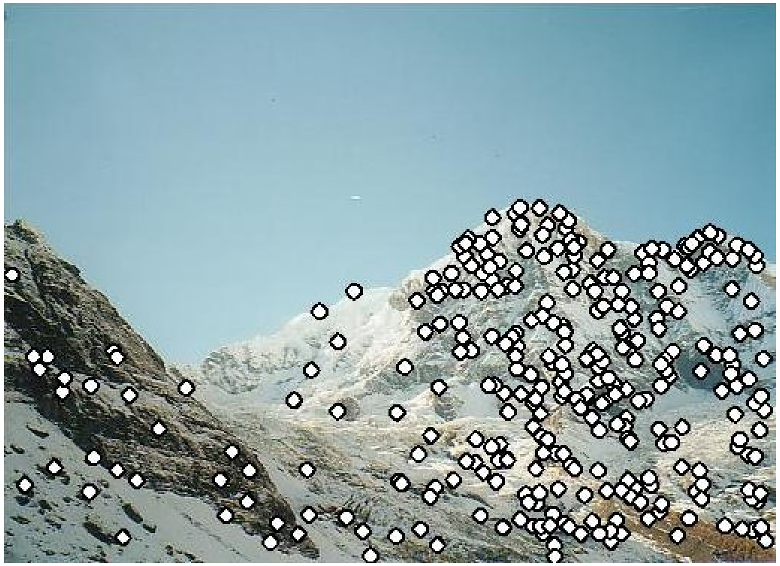
M. Brown and D. G. Lowe. Recognising Panoramas. ICCV 2003

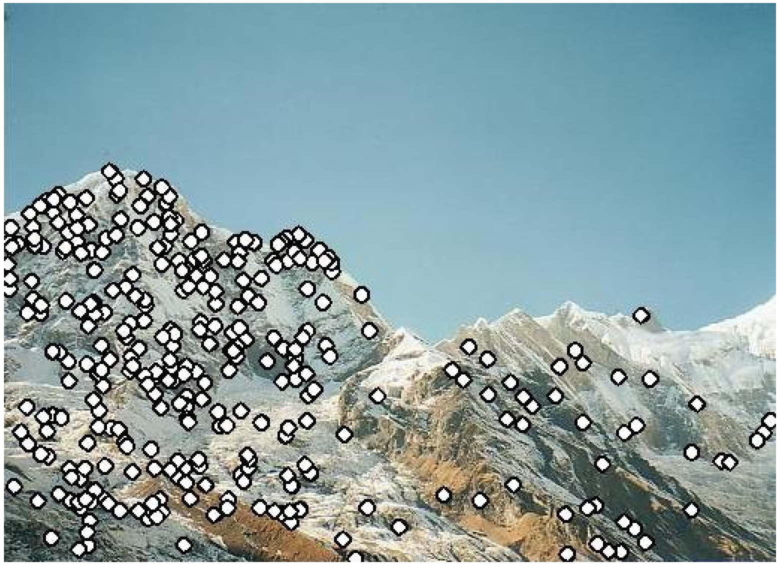
How do we build panorama?

- We need to match (align) images

Matching with Features

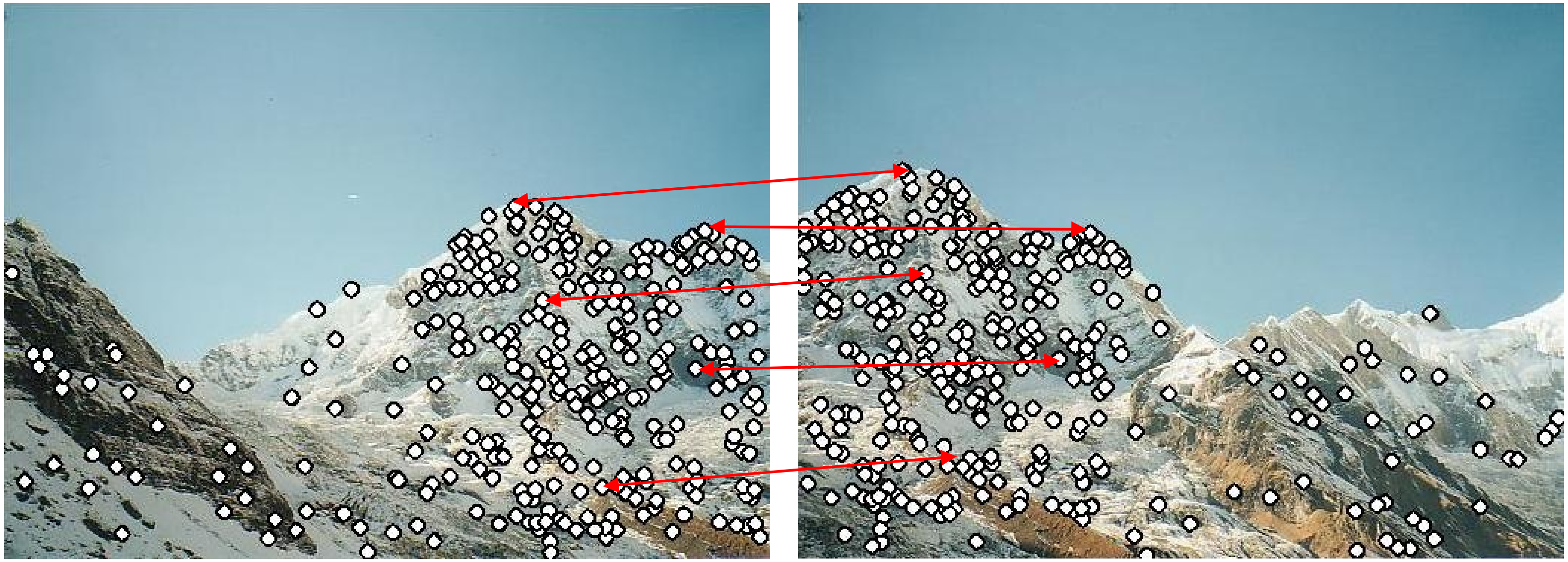
- Detect features (feature points) in both images





Matching with Features

- Detect features (feature points) in both images
- Match features - find corresponding pairs

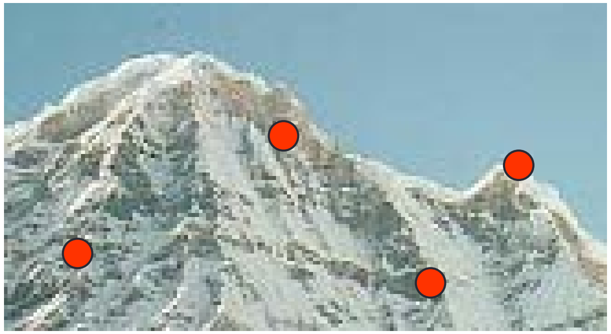


Matching with Features

- Detect features (feature points) in both images
- Match features - find corresponding pairs
- Use these pairs to align images

Matching with Features

- Problem 1:
  - Detect the *same* point *independently* in both images

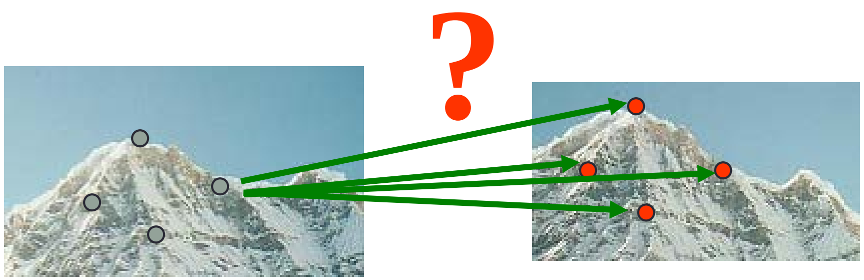


no chance to match!

We need a repeatable detector

Matching with Features

- Problem 2:
  - For each point correctly recognize the corresponding one

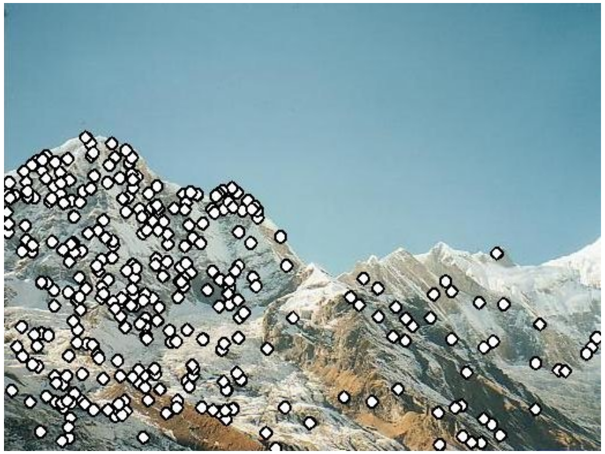


We need a reliable and distinctive ***descriptor***

More motivation…

- Feature points are used also for:
  - Image alignment (e.g. homography or *fundamental* matrix)
  - 3D reconstruction
  - Motion tracking
  - Object recognition
  - Indexing and database retrieval
  - Robot navigation
  - … other

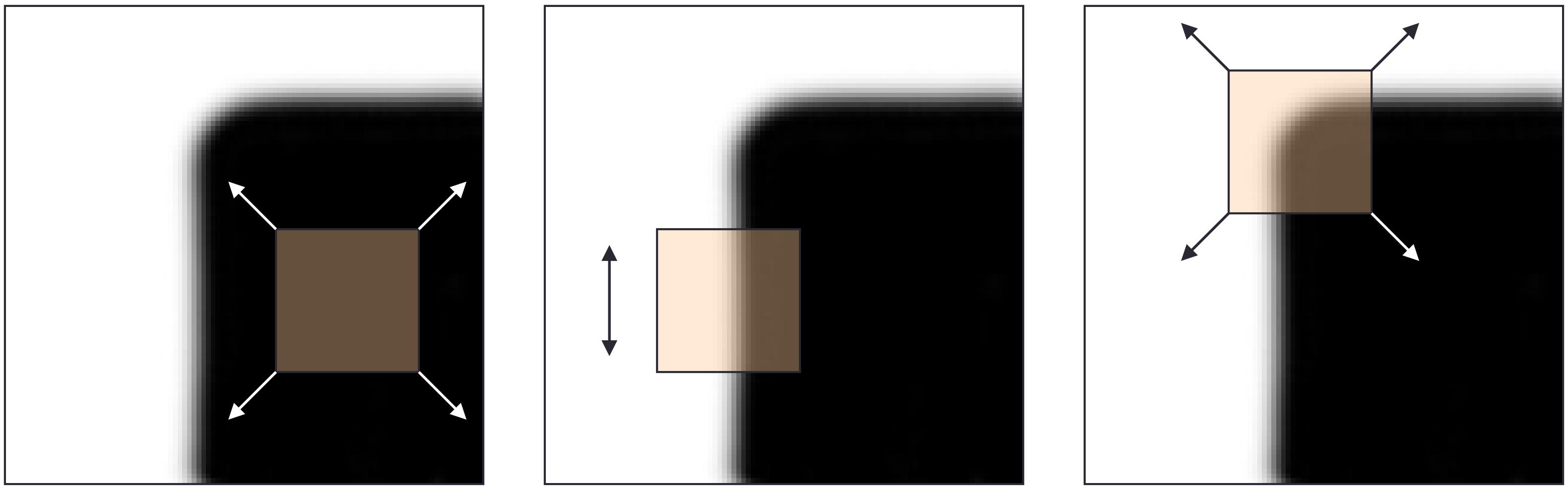
Characteristics of good features

- Repeatability/Precision
  - The same feature can be found in several images despite geometric and photometric transformations
- Saliency/Matchability
  - Each feature has a distinctive description
- Compactness and efficiency
  - Many fewer features than image pixels
- Locality
  - A feature occupies a relatively small area of the image; robust to clutter and occlusion

Corner Detection: Basic Idea

- We should easily recognize the point by looking through a small window
- Shifting a window in any direction should give a large change in intensity



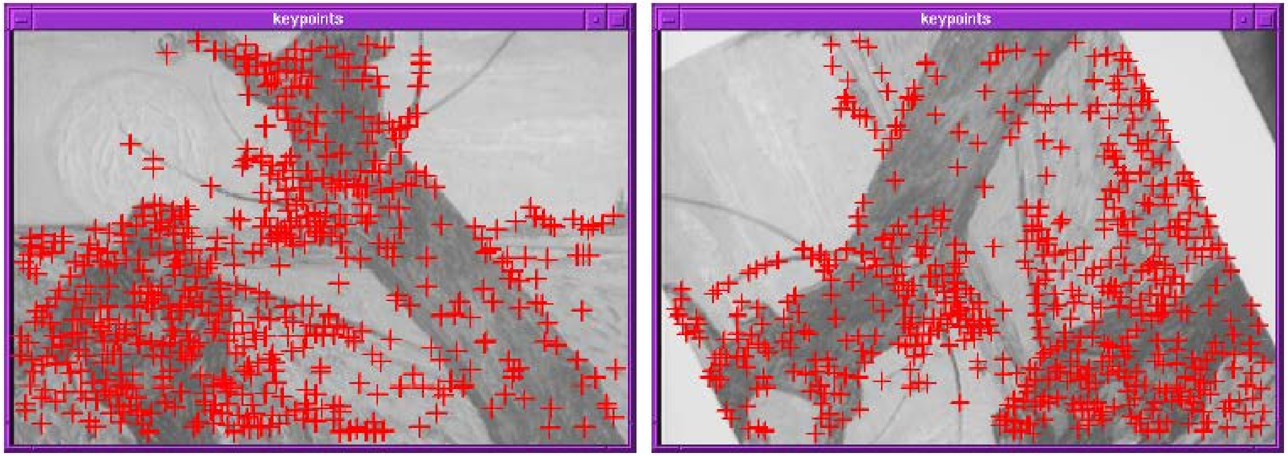
"flat" region:
no change in all directions

"edge":
no change along the edge direction

"corner":
significant change in all directions with small shift

Source: A. Efros

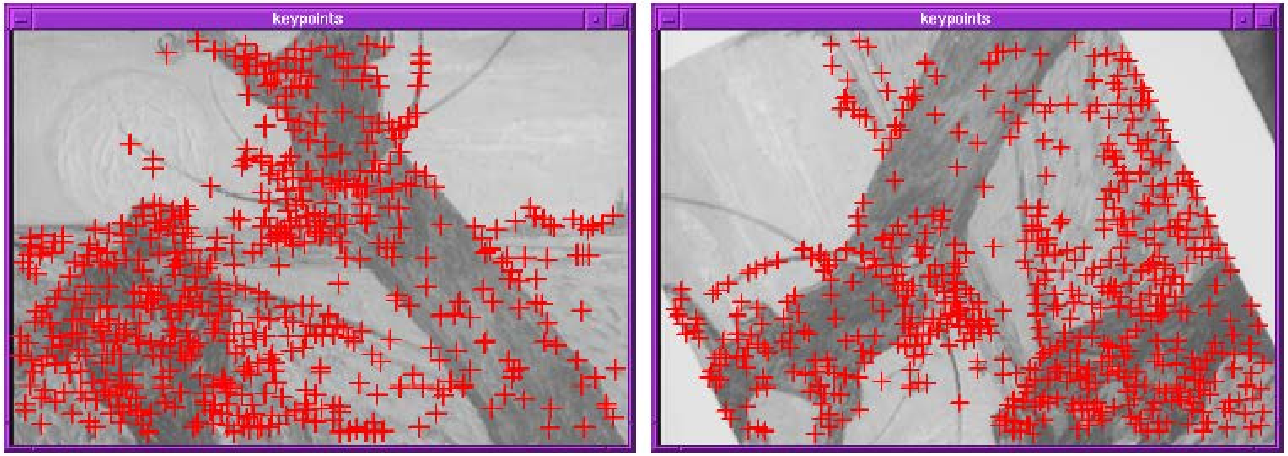
Finding *Corners*



- Key property: in the region around a corner, image gradient has two or more dominant directions
- Corners are repeatable and distinctive

C.Harris and M.Stephens. ["A Combined Corner and Edge Detector."](#) *Proceedings of the 4th Alvey Vision Conference*: pages 147—151, **1988**

Finding **Harris** *Corners*



- Key property: in the region around a corner, image gradient has two or more dominant directions
- Corners are repeatable and distinctive

C. **Harris** and M.Stephens. "A Combined Corner and Edge Detector.“ *Proceedings of the 4th Alvey Vision Conference*: pages 147—151, **1988**

Corner Detection: Mathematics

Change in appearance for the shift [*u,v*]:

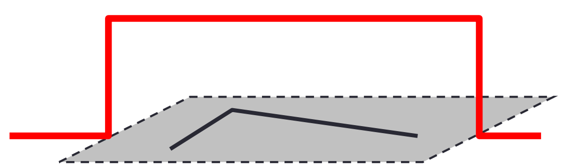
$$\boxed{E(u,v)=\sum_{x,y} w(x,y)\left[I(x+u,y+v)-I(x,y)\right]^2}$$

Window function

Shifted intensity

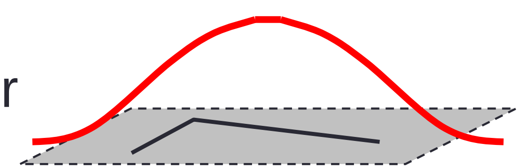
Intensity

Window function $w(x,y)$ =



1 in window, 0 outside

or



Gaussian

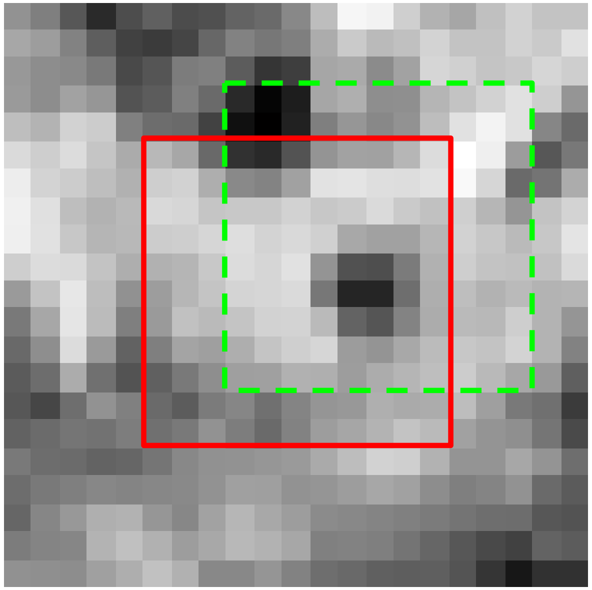
Source: R. Szeliski

Corner Detection: Mathematics

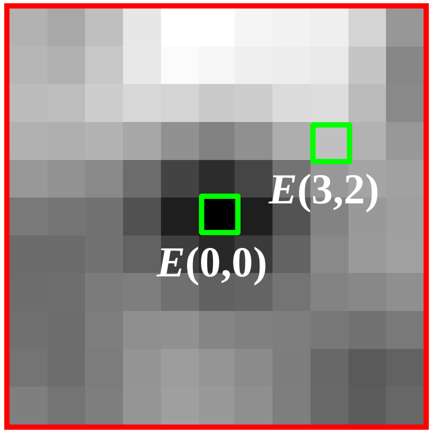
Change in appearance for the shift [*u,v*]:

$$E(u,v)=\sum_{x,y} w(x,y)\left[I(x+u,y+v)-I(x,y)\right]^2$$

$I(x, y)$



$E(u, v)$



Corner Detection: Mathematics

Change in appearance for the shift [*u,v*]:

$$E(u,v)=\sum_{x,y} w(x,y)\left[I(x+u,y+v)-I(x,y)\right]^2$$

We want to find out how this function behaves for ***small*** shifts (u,v near 0,0)

Corner Detection: Mathematics

Change in appearance for the shift [*u*,*v*]:

$$\boxed{E(u,v)=\sum_{x,y} w(x,y)\left[I(x+u,y+v)-I(x,y)\right]^2}$$

Second-order Taylor expansion of *E*(*u*,*v*) about (0,0) (local quadratic approximation for small *u*,*v*):

$$\text{(1D)}: F(\delta x)\approx F(0)+\delta x\cdot\frac{dF(0)}{dx}+\frac{1}{2}\delta x^2\cdot\frac{d^2F(0)}{dx^2}$$

$$E(u,v)\approx E(0,0)+[u\ \ v]\begin{bmatrix}E_u(0,0)\\ E_v(0,0)\end{bmatrix}+\frac{1}{2}[u\ \ v]\begin{bmatrix}E_{uu}(0,0) & E_{uv}(0,0)\\ E_{uv}(0,0) & E_{vv}(0,0)\end{bmatrix}\begin{bmatrix}u\\ v\end{bmatrix}$$

Corner Detection: Mathematics

$$E(u,v)=\sum_{x,y} w(x,y)\left[I(x+u,y+v)-I(x,y)\right]^2$$

Second-order Taylor expansion of $E(u,v)$ about (0,0):

$$E(u,v)\approx E(0,0)+[u\ \ v]\begin{bmatrix}E_u(0,0)\\ E_v(0,0)\end{bmatrix}+\frac{1}{2}[u\ \ v]\begin{bmatrix}E_{uu}(0,0) & E_{uv}(0,0)\\ E_{uv}(0,0) & E_{vv}(0,0)\end{bmatrix}\begin{bmatrix}u\\ v\end{bmatrix}$$

$$E_u(u,v)=\sum_{x,y} 2w(x,y)\left[I(x+u,y+v)-I(x,y)\right]I_x(x+u,y+v)$$

$$E_{uu}(u,v)=\sum_{x,y} 2w(x,y)I_x(x+u,y+v)I_x(x+u,y+v)$$
$$+\sum_{x,y} 2w(x,y)\left[I(x+u,y+v)-I(x,y)\right]I_{xx}(x+u,y+v)$$

$$E_{uv}(u,v)=\sum_{x,y} 2w(x,y)I_y(x+u,y+v)I_x(x+u,y+v)$$
$$+\sum_{x,y} 2w(x,y)\left[I(x+u,y+v)-I(x,y)\right]I_{xy}(x+u,y+v)$$

Corner Detection: Mathematics

$$\boxed{E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u,y+v) - I(x,y)\right]^2}$$

Second-order Taylor expansion of $E(u,v)$ about (0,0):

$$E(u,v) \approx E(0,0) + [u \quad v]\begin{bmatrix} E_u(0,0) \\ E_v(0,0) \end{bmatrix} + \frac{1}{2}[u \quad v]\begin{bmatrix} E_{uu}(0,0) & E_{uv}(0,0) \\ E_{uv}(0,0) & E_{vv}(0,0) \end{bmatrix}\begin{bmatrix} u \\ v \end{bmatrix}$$

$$E_u(u,v) = \sum_{x,y} 2w(x,y)\left[I(x+u,y+v) - I(x,y)\right] I_x(x+u,y+v)$$

$$E_{uu}(u,v) = \sum_{x,y} 2w(x,y) I_x(x+u,y+v) I_x(x+u,y+v)$$
$$+ \sum_{x,y} 2w(x,y)\left[I(x+u,y+v) - I(x,y)\right] I_{xx}(x+u,y+v)$$

$$E_{uv}(u,v) = \sum_{x,y} 2w(x,y) I_y(x+u,y+v) I_x(x+u,y+v)$$
$$+ \sum_{x,y} 2w(x,y)\left[I(x+u,y+v) - I(x,y)\right] I_{xy}(x+u,y+v)$$

Corner Detection: Mathematics

$$\boxed{E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u,y+v) - I(x,y)\right]^2}$$

Second-order Taylor expansion of $E(u,v)$ about (0,0):

$$E(u,v) \approx E(0,0) + [u \quad v]\begin{bmatrix} E_u(0,0) \\ E_v(0,0) \end{bmatrix} + \frac{1}{2}[u \quad v]\begin{bmatrix} E_{uu}(0,0) & E_{uv}(0,0) \\ E_{uv}(0,0) & E_{vv}(0,0) \end{bmatrix}\begin{bmatrix} u \\ v \end{bmatrix}$$

$$E_u(u,v) = \sum_{x,y} 2w(x,y)\left[I(x+u,y+v) - I(x,y)\right] I_x(x+u,y+v)$$

$$E_{uu}(u,v) = \sum_{x,y} 2w(x,y) I_x(x+u,y+v) I_x(x+u,y+v) + \sum_{x,y} 2w(x,y)\left[I(x+u,y+v) - I(x,y)\right] I_{xx}(x+u,y+v)$$

$$E_{uv}(u,v) = \sum_{x,y} 2w(x,y) I_y(x+u,y+v) I_x(x+u,y+v) + \sum_{x,y} 2w(x,y)\left[I(x+u,y+v) - I(x,y)\right] I_{xy}(x+u,y+v)$$

Corner Detection: Mathematics

$$\boxed{E(u,v)=\sum_{x,y} w(x,y)\left[I(x+u,y+v)-I(x,y)\right]^2}$$

Second-order Taylor expansion of $E(u,v)$ about (0,0):

$$E(u,v)\approx E(0,0)+[u\ \ v]\begin{bmatrix}E_u(0,0)\\ E_v(0,0)\end{bmatrix}+\frac{1}{2}[u\ \ v]\begin{bmatrix}E_{uu}(0,0) & E_{uv}(0,0)\\ E_{uv}(0,0) & E_{vv}(0,0)\end{bmatrix}\begin{bmatrix}u\\ v\end{bmatrix}$$

$$E_u(u,v)=\sum_{x,y} 2w(x,y)\left[I(x+u,y+v)-I(x,y)\right]I_x(x+u,y+v)$$

$$E_{uu}(u,v)=\sum_{x,y} 2w(x,y)I_x(x+u,y+v)I_x(x+u,y+v)$$

$$+\sum_{x,y} 2w(x,y)\left[I(x+u,y+v)-I(x,y)\right]I_{xx}(x+u,y+v)$$

$$E_{uv}(u,v)=\sum_{x,y} 2w(x,y)I_y(x+u,y+v)I_x(x+u,y+v)$$

$$+\sum_{x,y} 2w(x,y)\left[I(x+u,y+v)-I(x,y)\right]I_{xy}(x+u,y+v)$$

Corner Detection: Mathematics

$$\boxed{E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u,y+v) - I(x,y)\right]^2}$$

Evaluate at (u,v) = **(0,0)**:

$$E(u,v) \approx \underbrace{\boxed{E(0,0)}}^{=\,0} + [u \quad v]\begin{bmatrix} E_u(0,0) \\ E_v(0,0) \end{bmatrix} + \frac{1}{2}[u \quad v]\begin{bmatrix} E_{uu}(0,0) & E_{uv}(0,0) \\ E_{uv}(0,0) & E_{vv}(0,0) \end{bmatrix}\begin{bmatrix} u \\ v \end{bmatrix}$$

$$E_u(u,v) = \sum_{x,y} 2w(x,y)\underbrace{\boxed{\left[I(x+u,y+v) - I(x,y)\right]}}_{=\,0} I_x(x+u,y+v)$$

$$E_{uu}(u,v) = \sum_{x,y} 2w(x,y) I_x(x+u,y+v) I_x(x+u,y+v)$$
$$+ \sum_{x,y} 2w(x,y)\underbrace{\boxed{\left[I(x+u,y+v) - I(x,y)\right]}}_{=\,0} I_{xx}(x+u,y+v)$$

$$E_{uv}(u,v) = \sum_{x,y} 2w(x,y) I_y(x+u,y+v) I_x(x+u,y+v)$$
$$+ \sum_{x,y} 2w(x,y)\underbrace{\boxed{\left[I(x+u,y+v) - I(x,y)\right]}}_{=\,0} I_{xy}(x+u,y+v)$$

Corner Detection: Mathematics

$$\boxed{E(u,v)=\sum_{x,y}w(x,y)\left[I(x+u,y+v)-I(x,y)\right]^2}$$

Second-order Taylor expansion of $E(u,v)$ about (0,0):

$$E(u,v)\approx E(0,0)+[u\ \ v]\begin{bmatrix}E_u(0,0)\\ E_v(0,0)\end{bmatrix}+\boxed{\frac{1}{2}[u\ \ v]\begin{bmatrix}E_{uu}(0,0) & E_{uv}(0,0)\\ E_{uv}(0,0) & E_{vv}(0,0)\end{bmatrix}\begin{bmatrix}u\\ v\end{bmatrix}}$$

$$E(0,0)=0$$

$$E_u(0,0)=0$$

$$E_v(0,0)=0$$

$$E_{uu}(0,0)=\sum_{x,y}2w(x,y)I_x(x,y)I_x(x,y)$$

$$E_{vv}(0,0)=\sum_{x,y}2w(x,y)I_y(x,y)I_y(x,y)$$

$$E_{uv}(0,0)=\sum_{x,y}2w(x,y)I_x(x,y)I_y(x,y)$$

Corner Detection: Mathematics

$$\boxed{E(u,v) = \sum_{x,y} w(x,y)\left[I(x+u, y+v) - I(x,y)\right]^2}$$

Second-order Taylor expansion of *E*(*u*,*v*) about (0,0):

$$E(u,v) \approx [u \;\; v] \begin{bmatrix} \sum_{x,y} w(x,y) I_x^2(x,y) & \sum_{x,y} w(x,y) I_x(x,y) I_y(x,y) \\ \sum_{x,y} w(x,y) I_x(x,y) I_y(x,y) & \sum_{x,y} w(x,y) I_y^2(x,y) \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$

$$E(0,0) = 0$$

$$E_u(0,0) = 0$$

$$E_v(0,0) = 0$$

$$E_{uu}(0,0) = \sum_{x,y} 2w(x,y) I_x(x,y) I_x(x,y)$$

$$E_{vv}(0,0) = \sum_{x,y} 2w(x,y) I_y(x,y) I_y(x,y)$$

$$E_{uv}(0,0) = \sum_{x,y} 2w(x,y) I_x(x,y) I_y(x,y)$$

Corner Detection: Mathematics

The quadratic approximation simplifies to

$$E(u,v) \approx [u \;\; v] \; M \begin{bmatrix} u \\ v \end{bmatrix}$$

where *M* is a *second moment matrix* computed from image derivatives:

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

Each product is a rank 1 2x2

Without weight

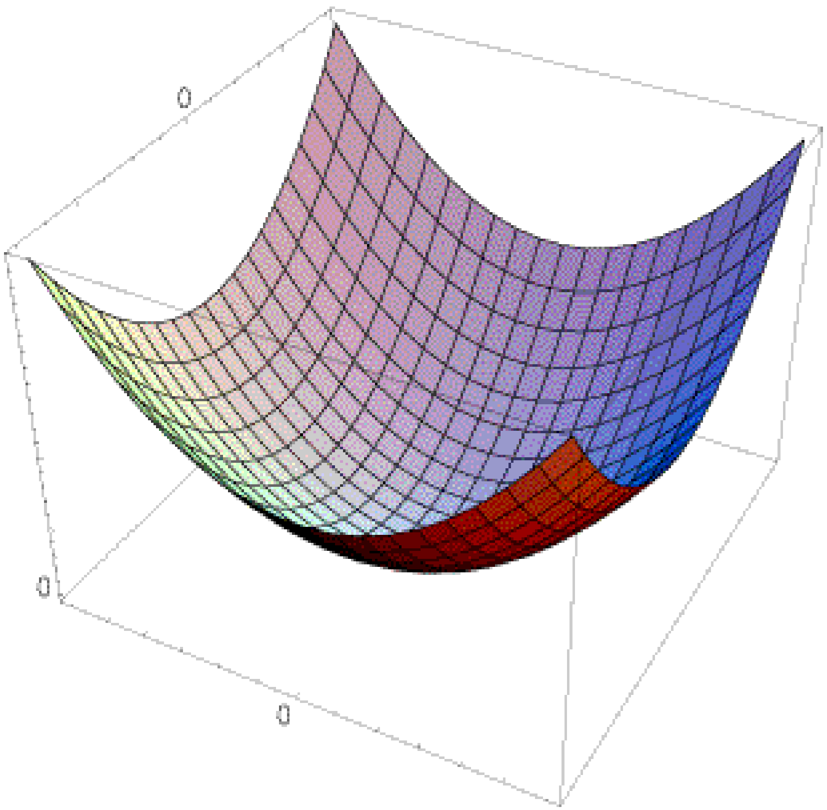
$$M = \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} = \sum \begin{bmatrix} I_x \\ I_y \end{bmatrix} [I_x \; I_y] = \sum \nabla I (\nabla I)^T$$

Interpreting the second moment matrix

The surface $E(u,v)$ is locally approximated by a quadratic form.

$$E(u,v) \approx [u \;\; v] \; M \begin{bmatrix} u \\ v \end{bmatrix}$$

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}$$

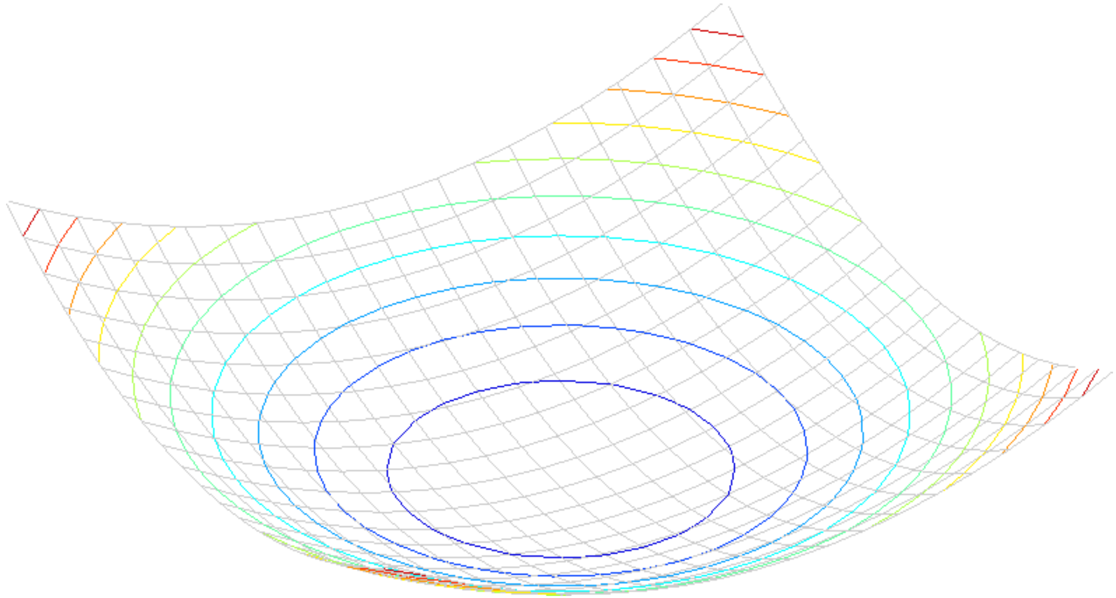


Interpreting the second moment matrix

Consider a constant "slice" of $E(u, v)$: $[u \;\; v] \; M \begin{bmatrix} u \\ v \end{bmatrix} = \text{const}$

$$I_x^2 u^2 + 2 I_x I_y uv + I_y^2 v^2 = k$$

This is the equation of an ellipse.



Interpreting the second moment matrix

First, consider the axis-aligned case where gradients are either horizontal or vertical

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

If either λ is close to 0, then this is **not** a corner, so look for locations where both are large.

Interpreting the second moment matrix

First, consider the axis-aligned case where gradients are either horizontal or vertical

$$M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix}$$

If either λ is close to 0, then this is **not** a corner, so look for locations where both are large.

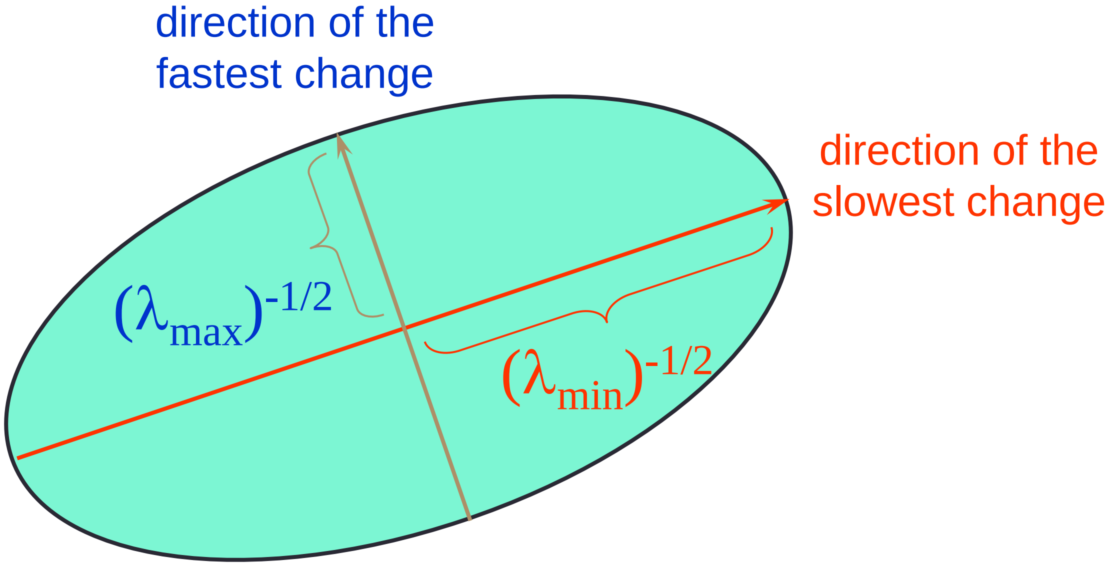
Interpreting the second moment matrix

Consider a horizontal "slice" of $E(u, v)$: $$[u\ \ v]\ M \begin{bmatrix} u \\ v \end{bmatrix} = \text{const}$$

This is the equation of an ellipse.

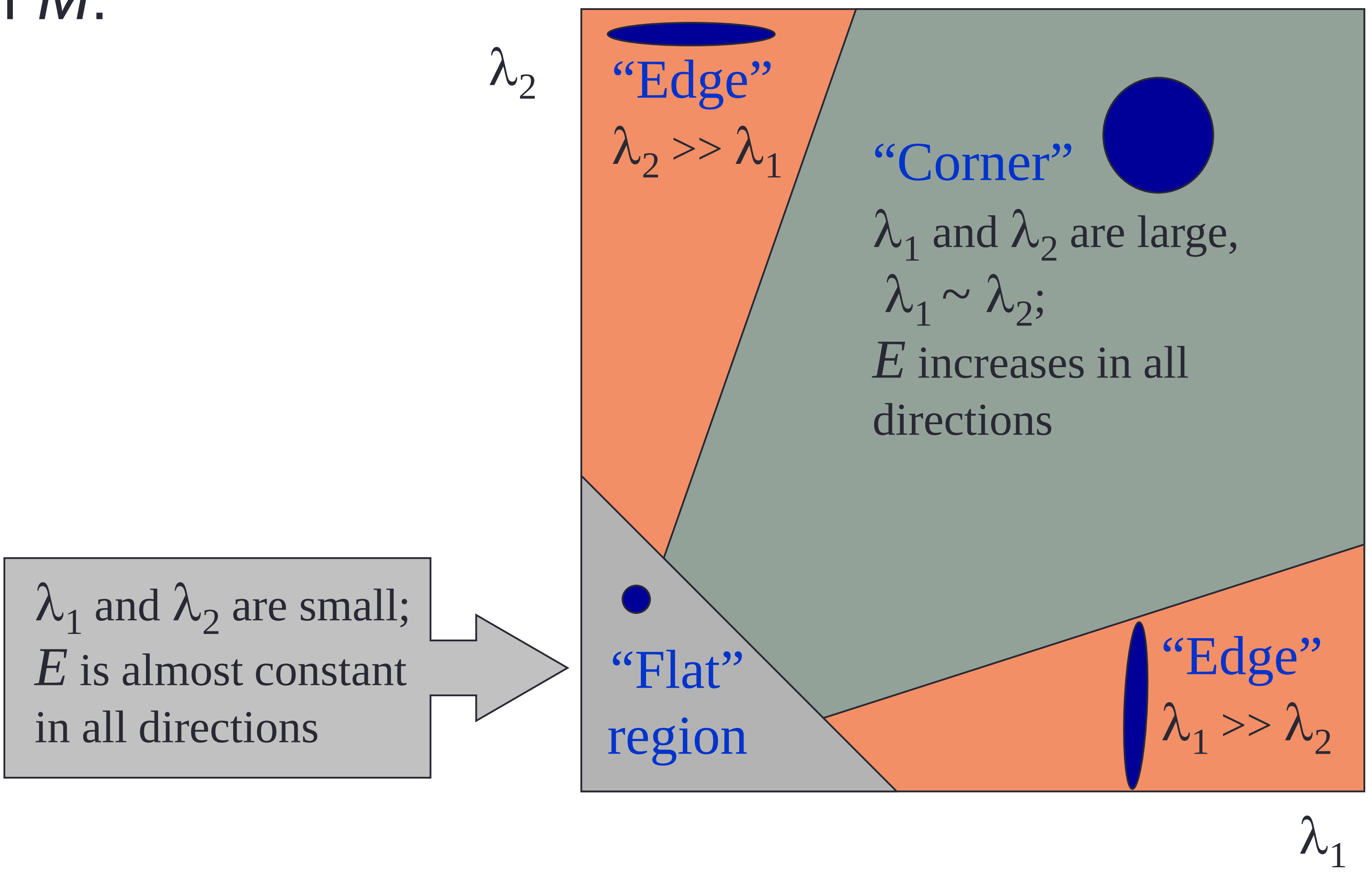
Diagonalization of M: $$M = R^{-1} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} R$$

The axis lengths of the ellipse are determined by the eigenvalues and the orientation is determined by *R*



Interpreting the eigenvalues

Classification of image points using eigenvalues of *M*:

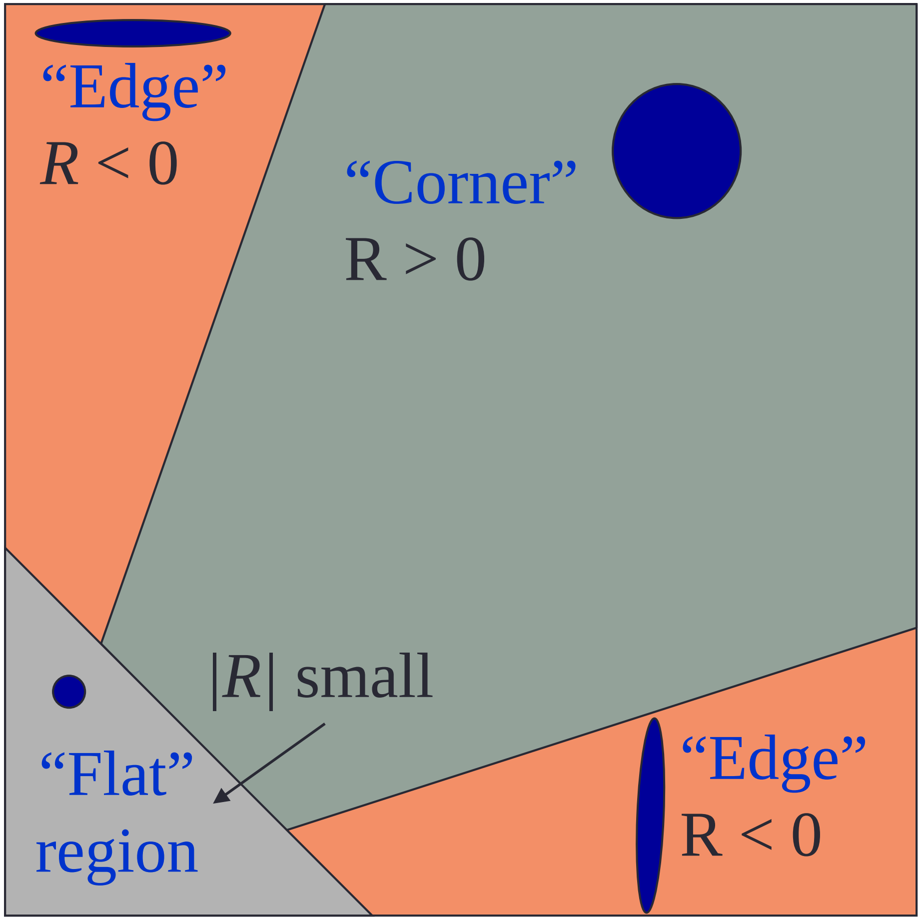


Harris corner response function

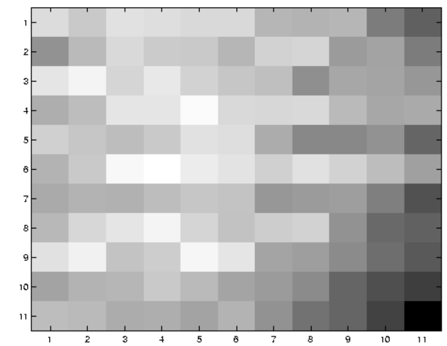
$$R = \det(M) - \alpha \operatorname{trace}(M)^2 = \lambda_1 \lambda_2 - \alpha(\lambda_1 + \lambda_2)^2$$

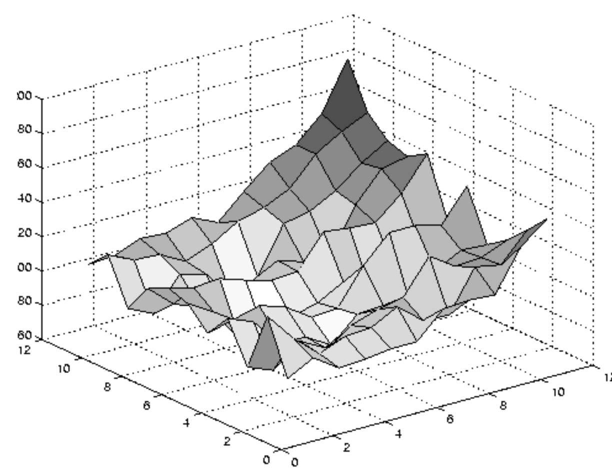
α: constant (0.04 to 0.06)

- R depends only on eigenvalues of M, but don't compute them (no sqrt, so really fast!
- R is large for a corner
- R is negative with large magnitude for an edge
- $|R|$ is small for a flat region



Low texture region



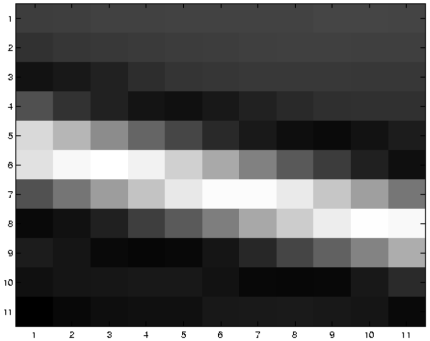


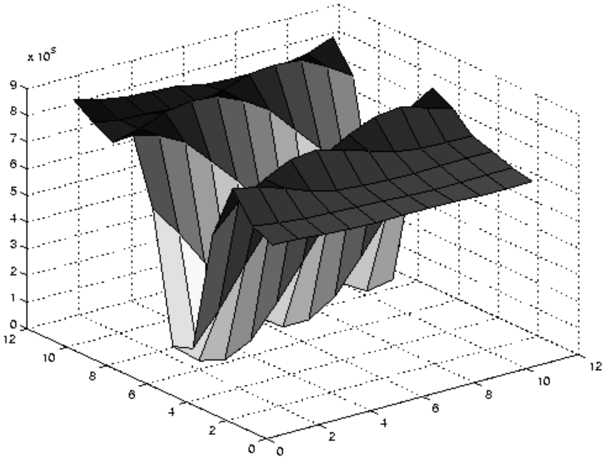
$$\sum \nabla I (\nabla I)^T$$

– gradients have small magnitude

– small λ_1, small λ_2

Edge



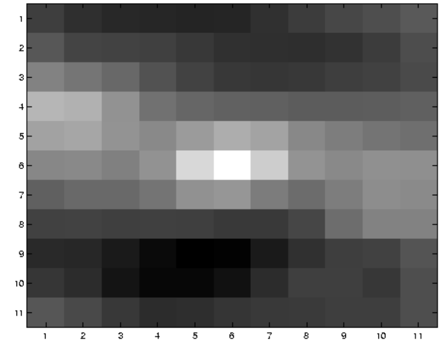


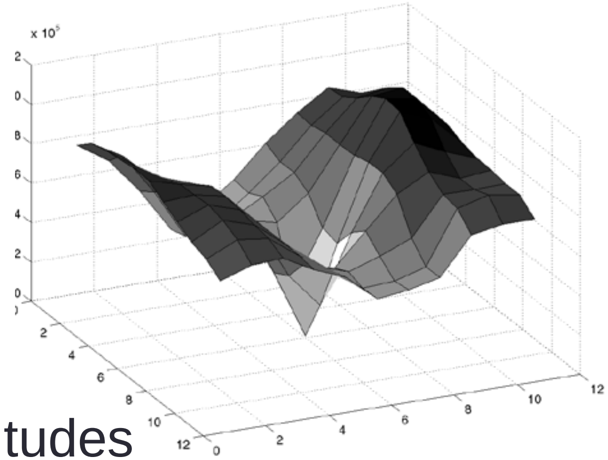
$$\sum \nabla I (\nabla I)^T$$

– large gradients, all the same

– large λ_1, small λ_2

High textured region





$$\sum \nabla I (\nabla I)^T$$

– gradients are different, large magnitudes

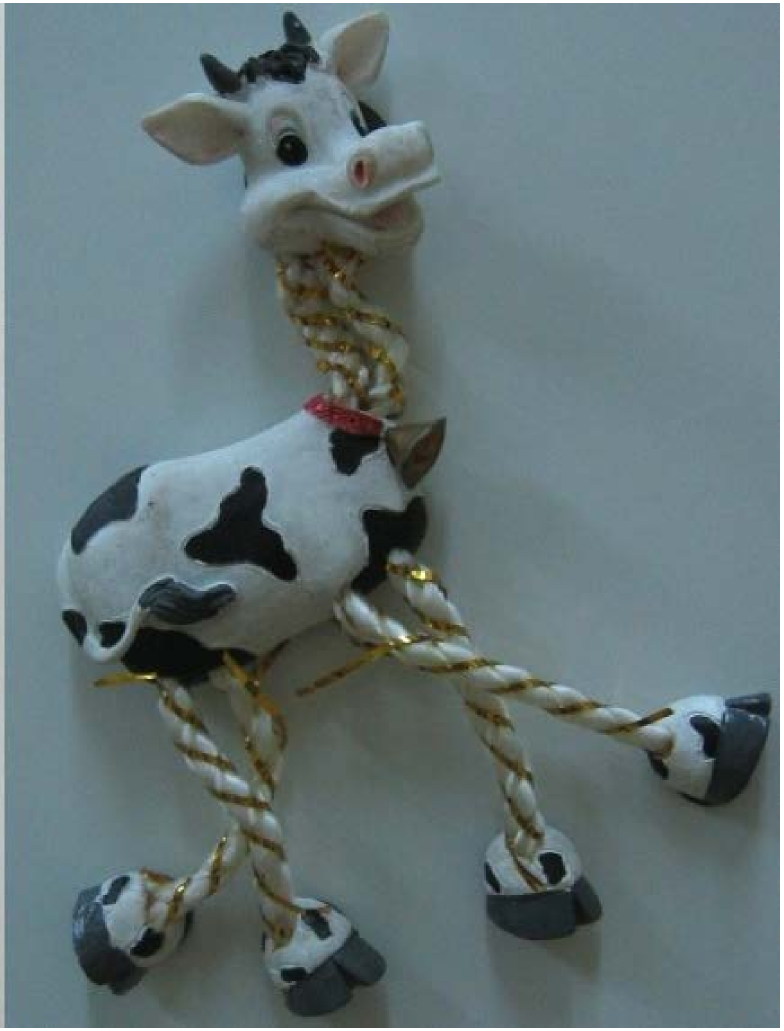
– large λ_1, large λ_2

Harris detector: Algorithm

1. Compute Gaussian derivatives at each pixel
2. Compute second moment matrix *M* in a Gaussian window around each pixel
3. Compute corner response function *R*
4. Threshold *R*
5. Find local maxima of response function (nonmaximum suppression)

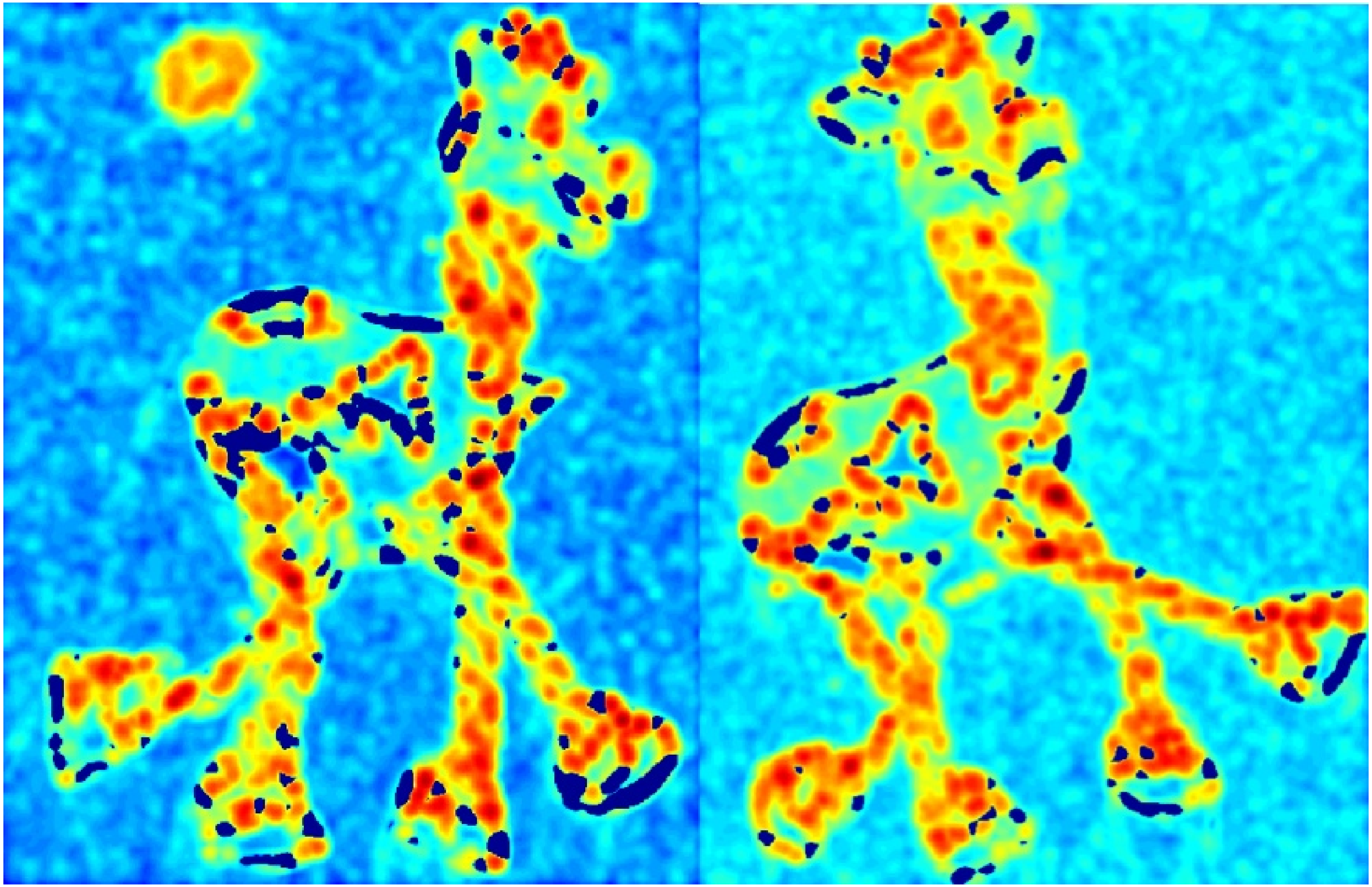
C.Harris and M.Stephens. "A Combined Corner and Edge Detector.“ *Proceedings of the 4th Alvey Vision Conference*: pages 147—151, 1988.

Harris Detector: Workflow



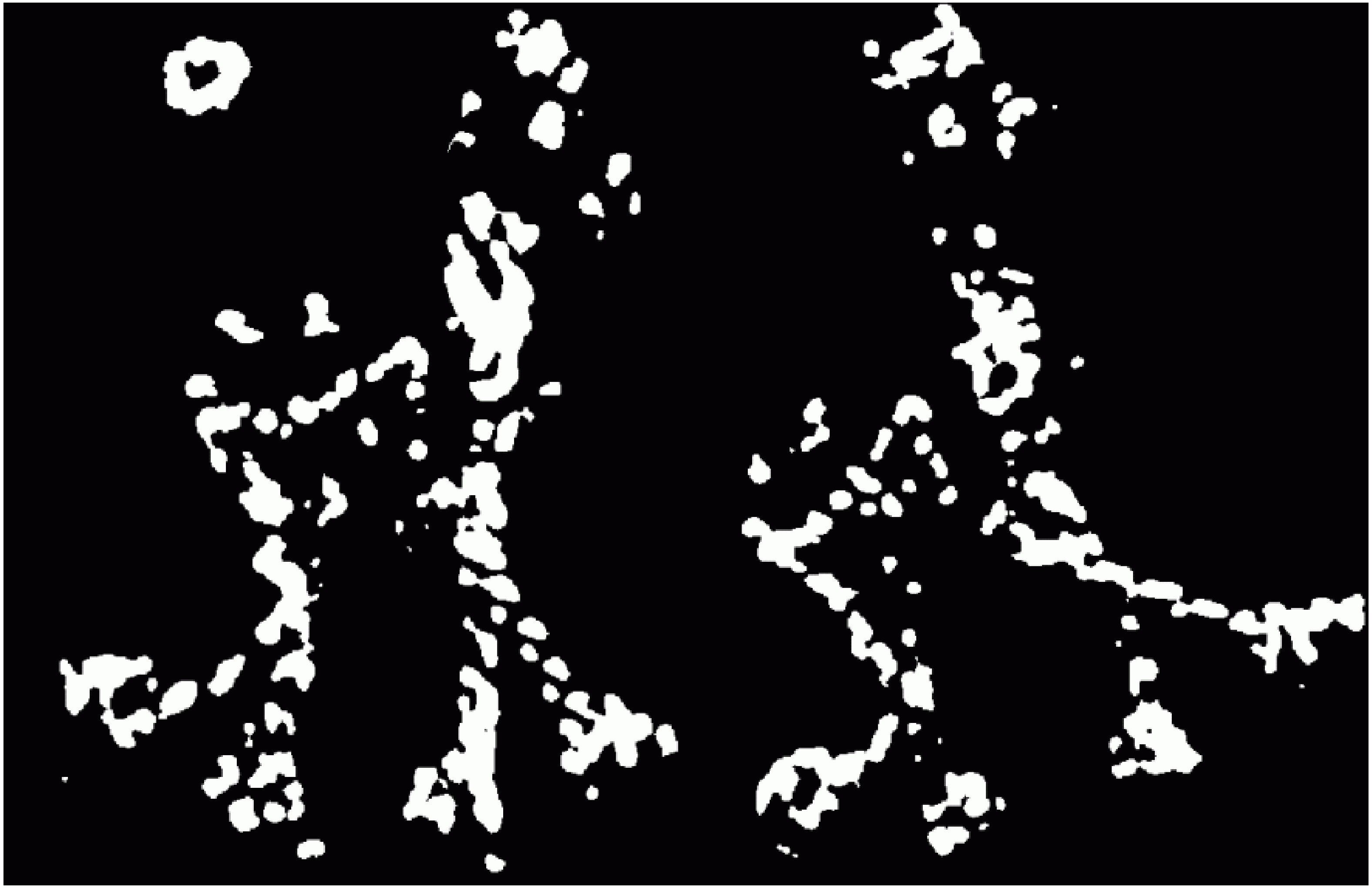
Harris Detector: Workflow

Compute corner response R



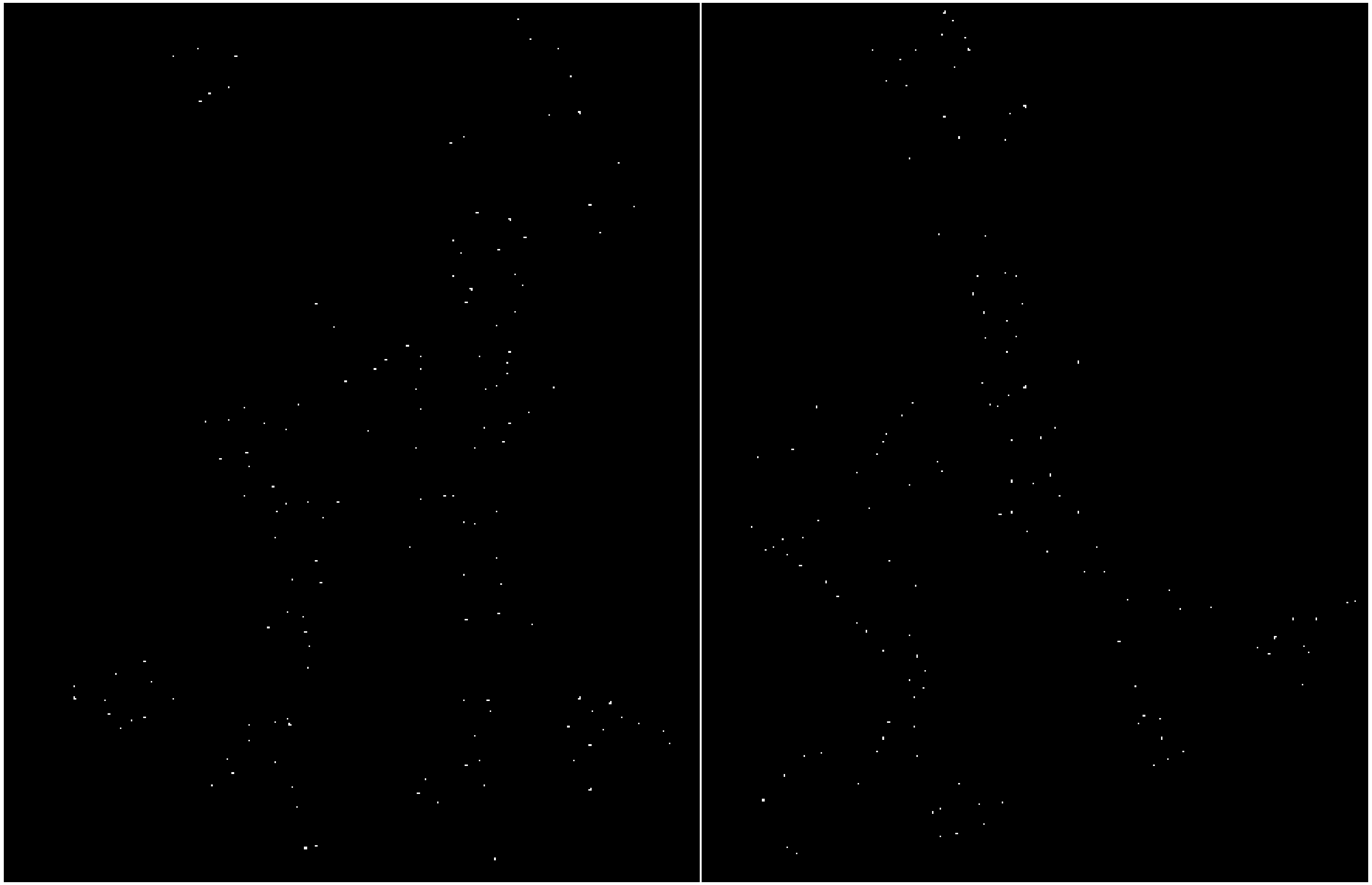
Harris Detector: Workflow

Find points with large corner response: R>threshold



Harris Detector: Workflow

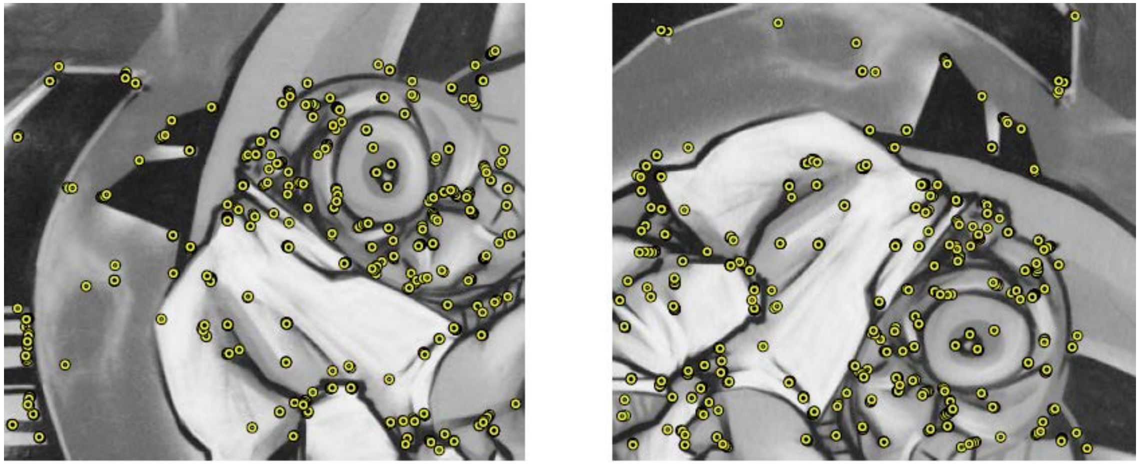
Take only the points of local maxima of R



Harris Detector: Workflow

Other corners:

- Shi-Tomasi '94:
  - "Cornerness" = min (λ_1, λ_2) Find local maximums
  - cvGoodFeaturesToTrack(...)
  - Reportedly better for region undergoing affine deformations

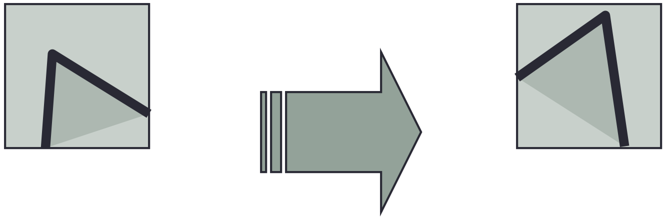


- Brown, M., Szeliski, R., and Winder, S. (2005): $\frac{\det M}{\operatorname{tr} M} = \frac{\lambda_0 \lambda_1}{\lambda_0 + \lambda_1}$
- there are others…

Harris Detector: Some Properties

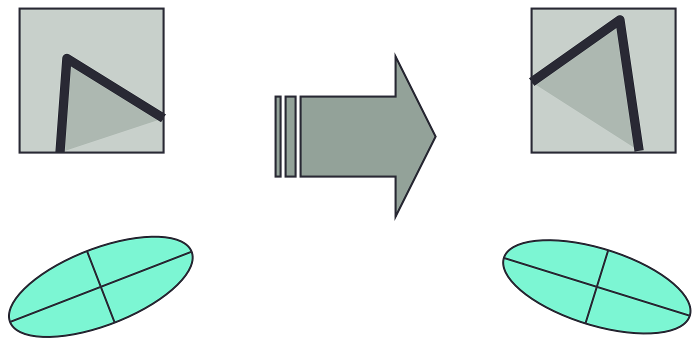
Harris Detector: Some Properties

- Rotation invariance?



Harris Detector: Some Properties

- Rotation invariance



Ellipse rotates but its shape (i.e. eigenvalues) remains the same

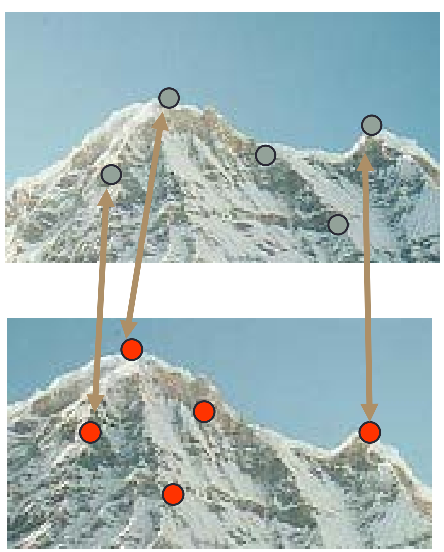
Corner response R is invariant to image rotation

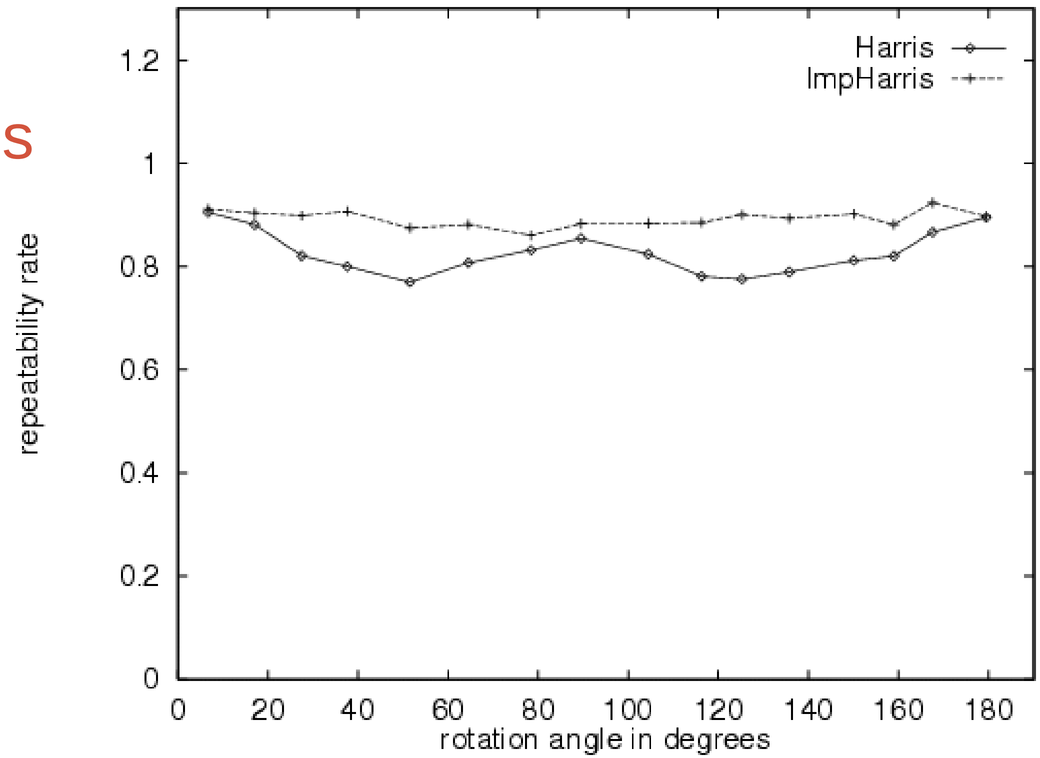
Rotation Invariant Detection

Harris Corner Detector

Repeatability rate:

$$\frac{\text{\# correspondences}}{\text{\# possible correspondences}}$$





C.Schmid et.al. “Evaluation of Interest Point Detectors”. IJCV 2000

Harris Detector: Some Properties

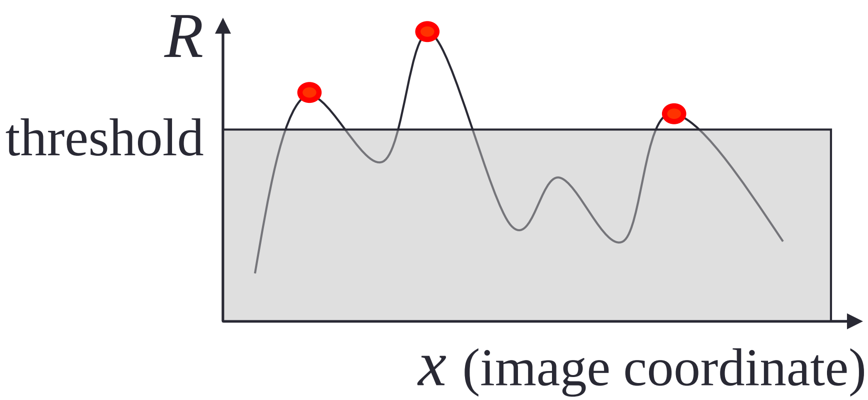
- Invariance to image intensity change?

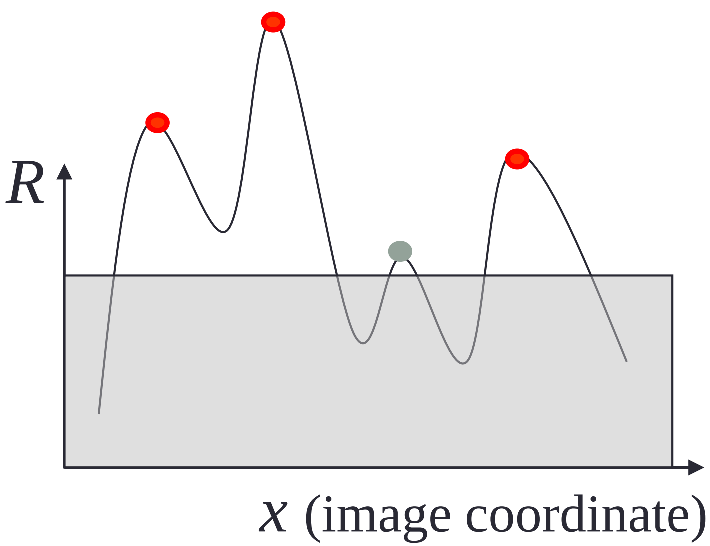
Harris Detector: Some Properties

- Partial invariance to additive and multiplicative intensity changes (threshold issue for multiplicative)

✓ Only derivatives are used => invariance to intensity shift $I \rightarrow I + b$

✓ Intensity scale: $I \rightarrow a\, I$



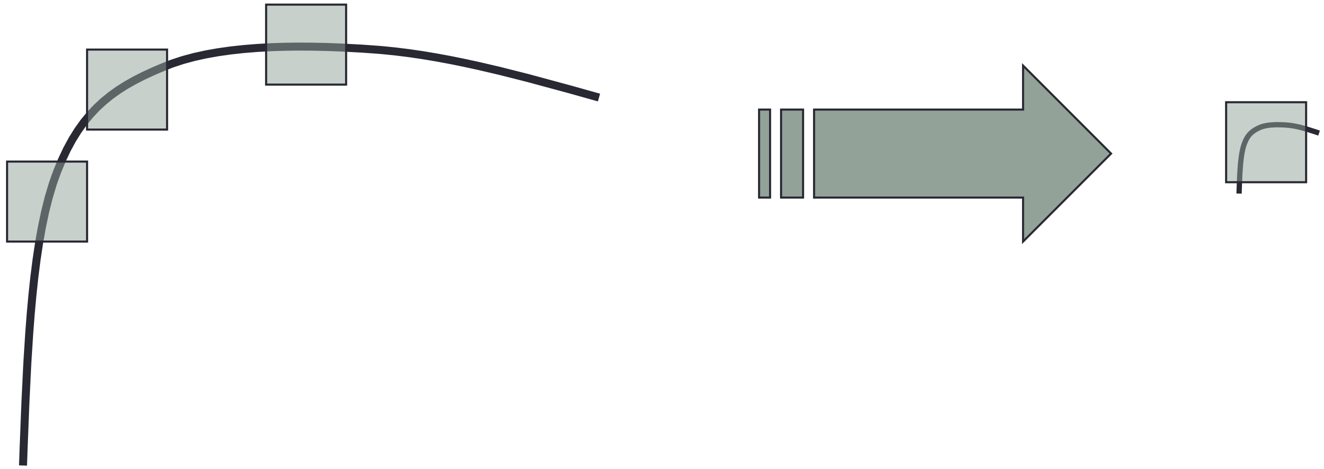


Harris Detector: Some Properties

- Invariant to image scale?

Harris Detector: Some Properties

- **Not invariant to *image scale*!**

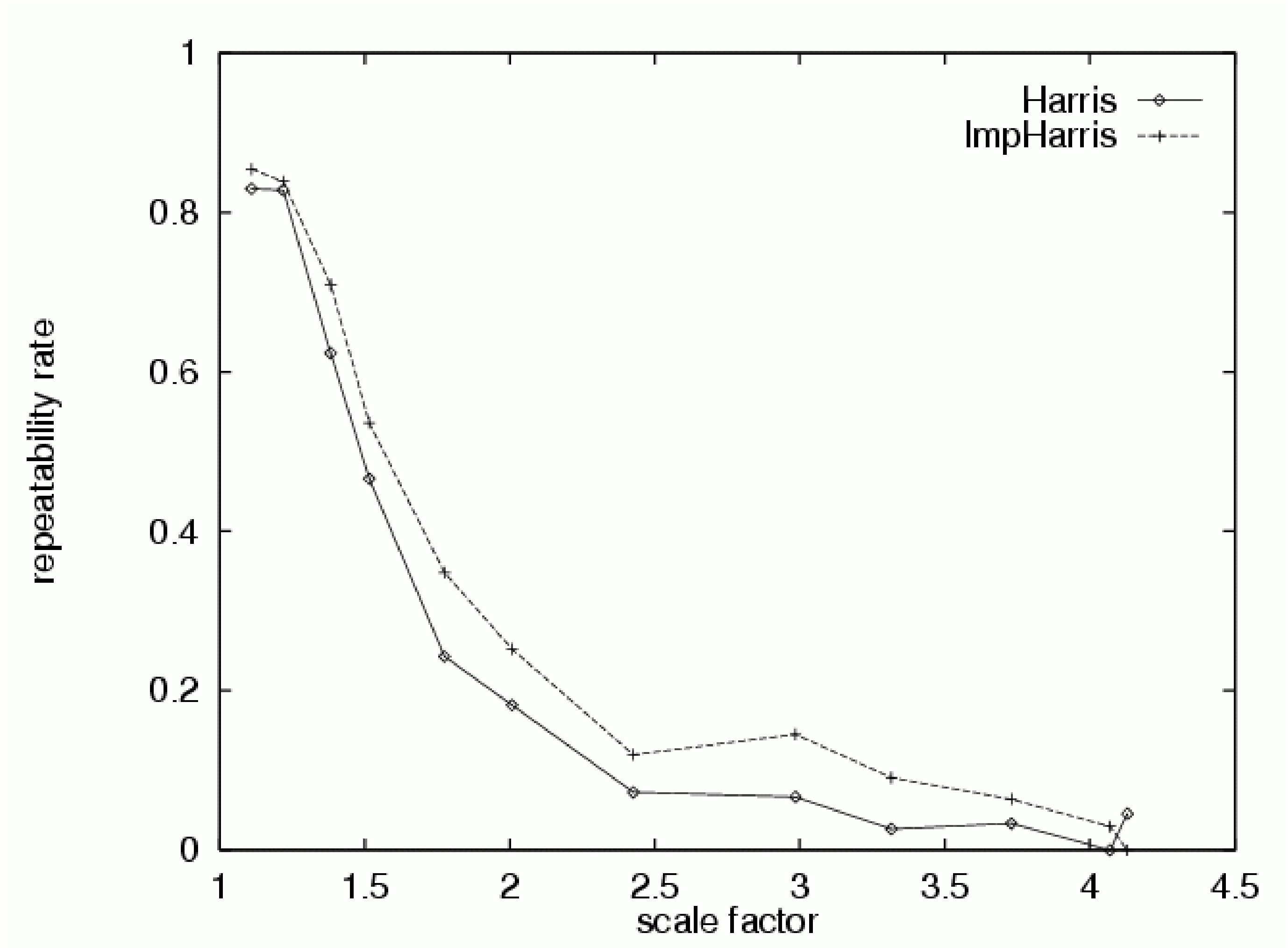


All points will be classified as edges

Corner !

Harris Detector: Some Properties

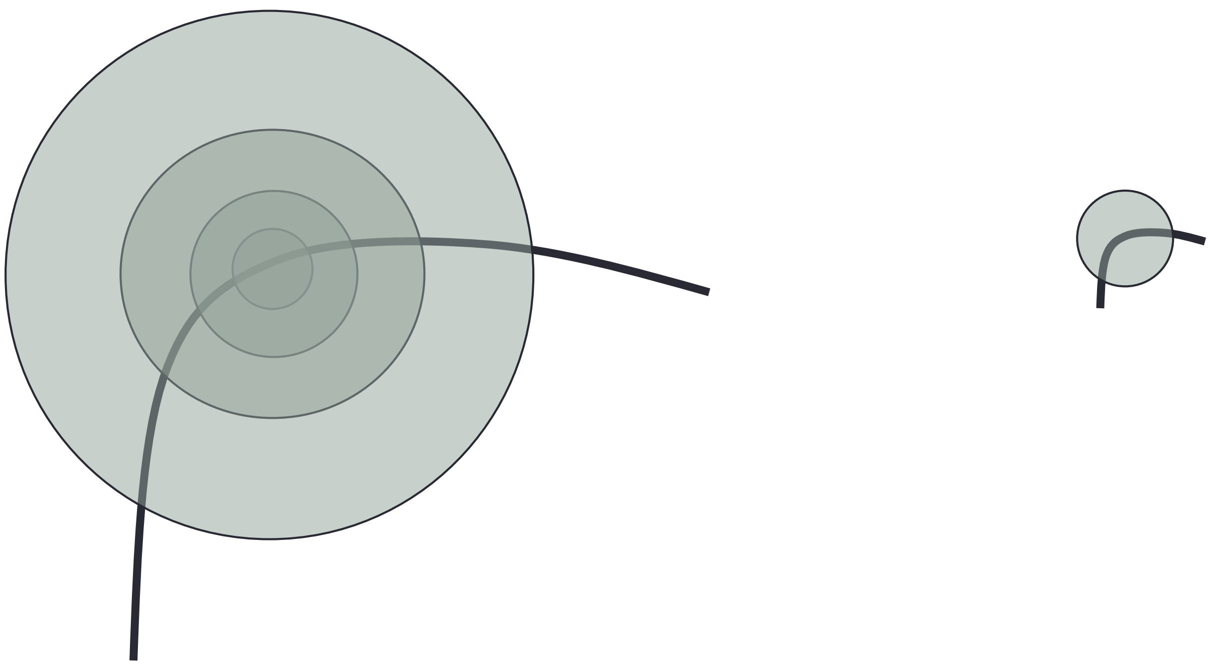
- Quality of Harris detector for different scale changes



IF we want scale invariance…

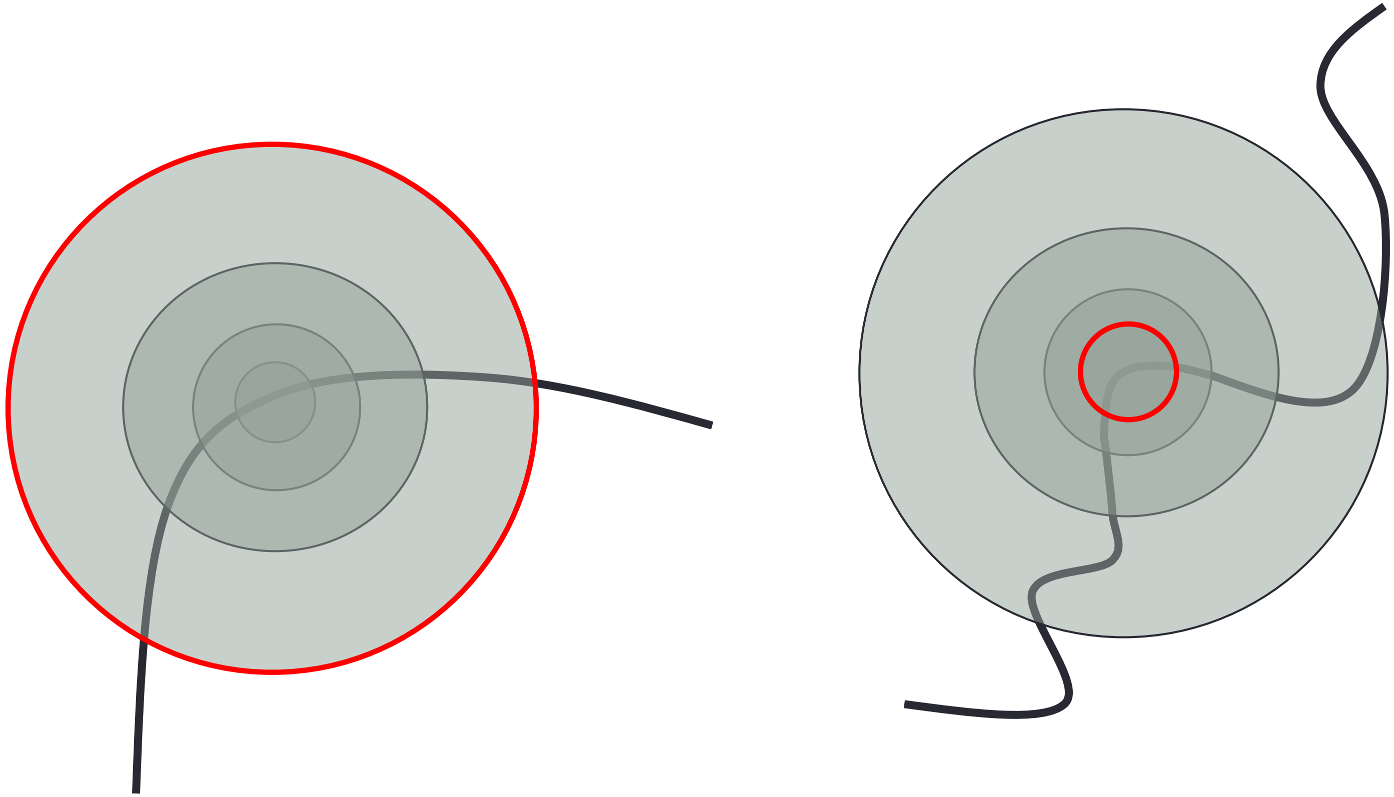
Scale Invariant Detection

- Consider regions (e.g. circles) of different sizes around a point
- Regions of corresponding sizes will look the same in both images



Scale Invariant Detection

- The problem: how do we choose corresponding circles independently in each image?

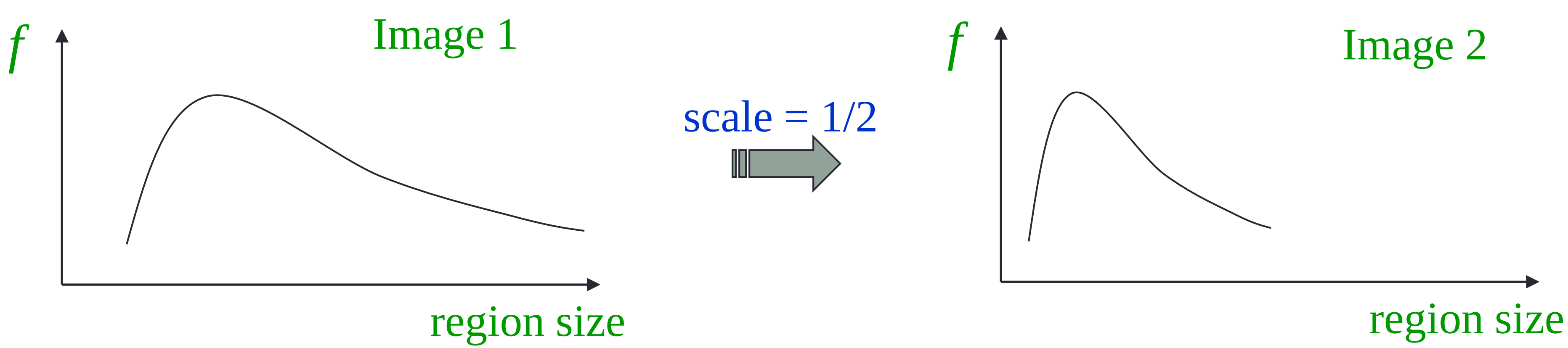


Scale Invariant Detection

- Solution:
  - Design a function on the region (circle), which is “scale invariant” (the same for corresponding regions, even if they are at different scales)

Example: average intensity. For corresponding regions (even of different sizes) it will be the same.

For some given point in one image, we can consider it as a function of region size (circle radius)



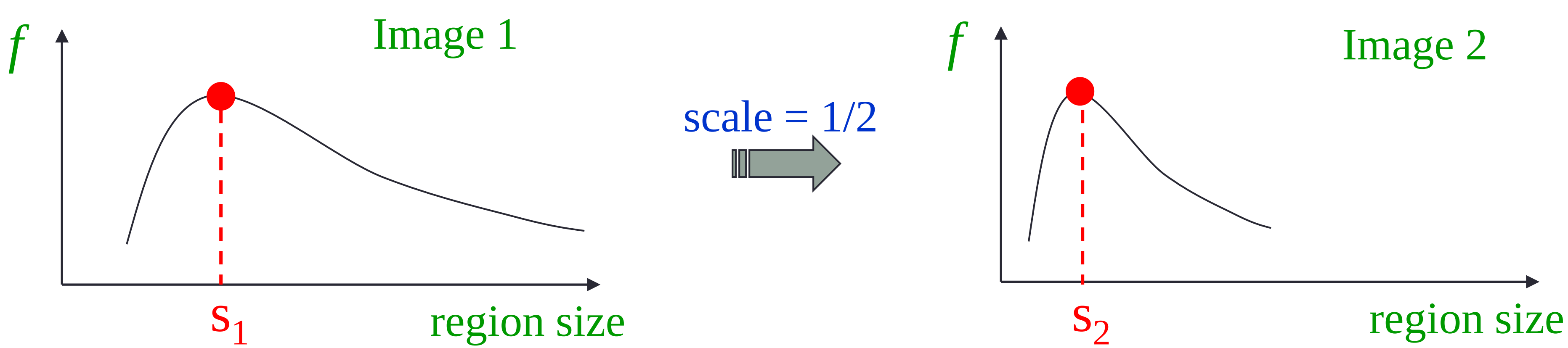
Scale Invariant Detection

- Common approach:

Take a local maximum of this function

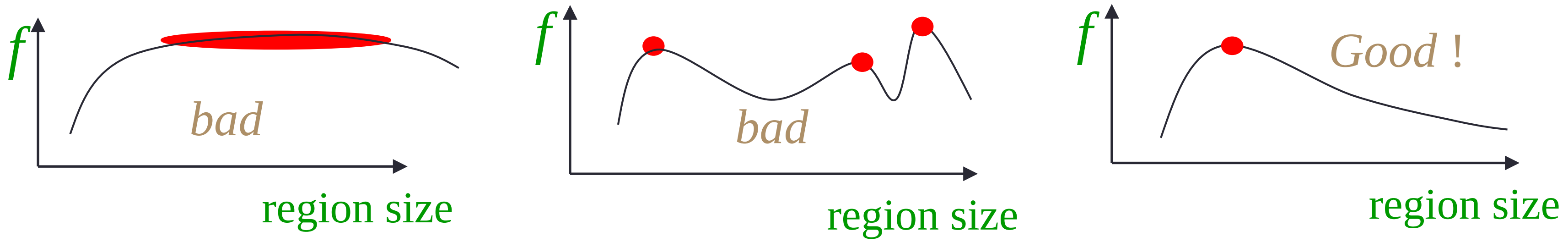
Observation: region size, for which the maximum is achieved, should be *invariant* to image scale.

Important: this scale invariant region size is found in each image independently!



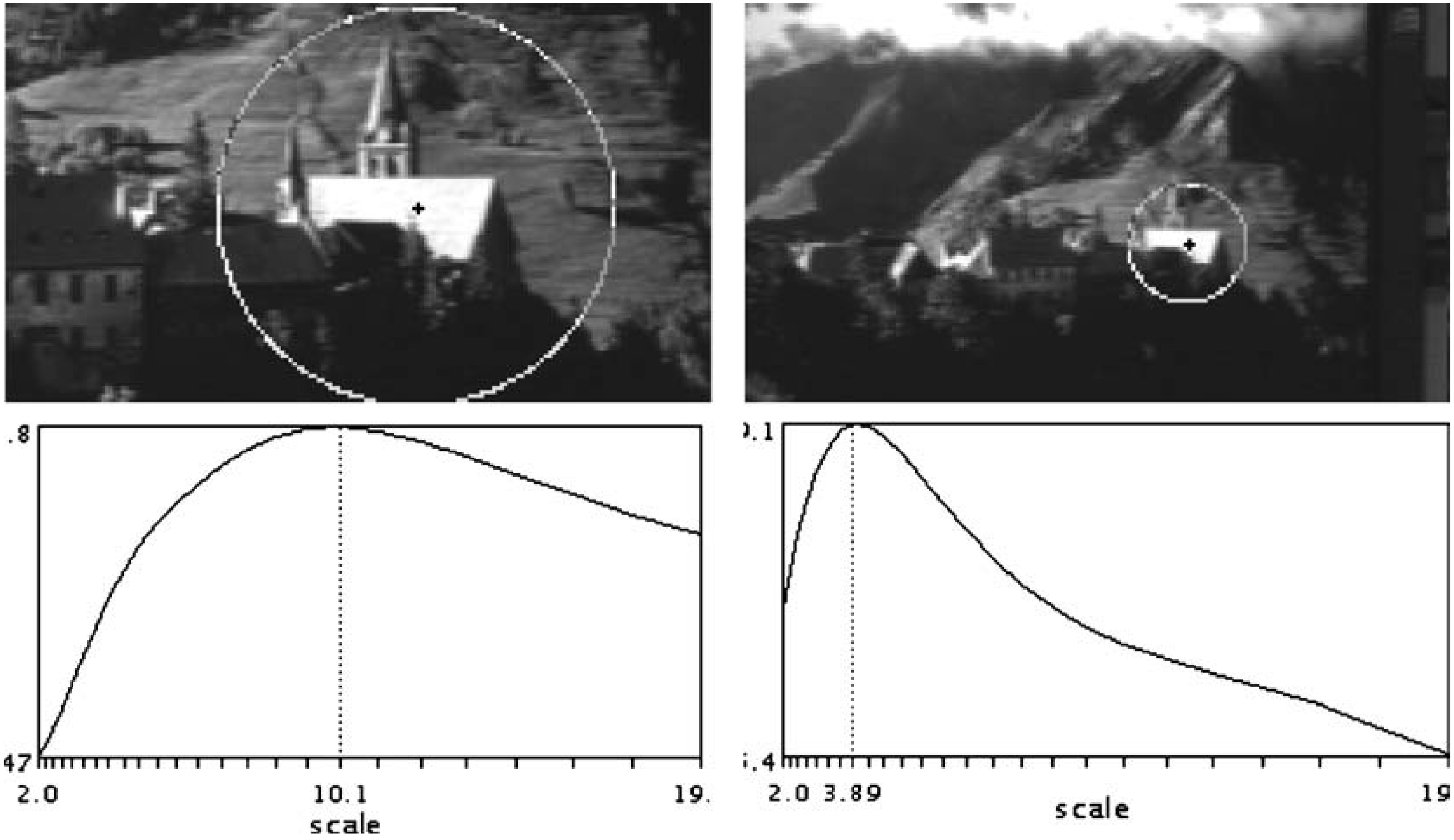
Scale Invariant Detection

- A “good” function for scale detection: has one stable sharp peak



- For usual images: a good function would be a one which responds to contrast (sharp local intensity change)

Scale sensitive response

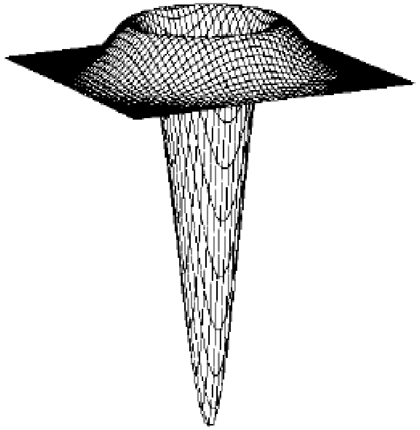


Scale Invariant Detection

Function is just application of a kernel: $f = \text{Kernel} * \text{Image}$

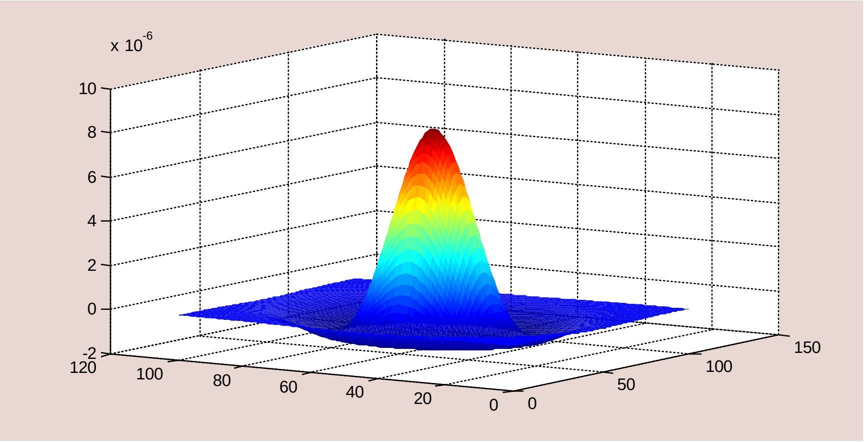
(Laplacian of Gaussian - LoG)

$$L = \sigma^2 \left(G_{xx}(x, y, \sigma) + G_{yy}(x, y, \sigma) \right)$$



$$\nabla^2 h_\sigma(u, v)$$

Laplacian of Gaussian



Scale Invariant Detection

- Functions for determining scale

$$f = \text{Kernel} * \text{Image}$$

Kernels:

$$L = \sigma^2 \left(G_{xx}(x, y, \sigma) + G_{yy}(x, y, \sigma) \right)$$

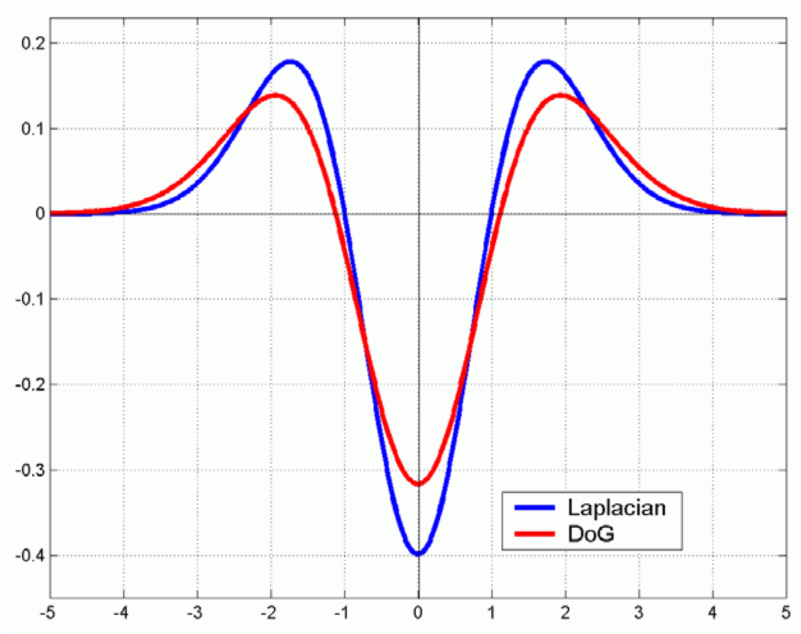
(Laplacian)

$$DoG = G(x, y, k\sigma) - G(x, y, \sigma)$$

(Difference of Gaussians)

where Gaussian

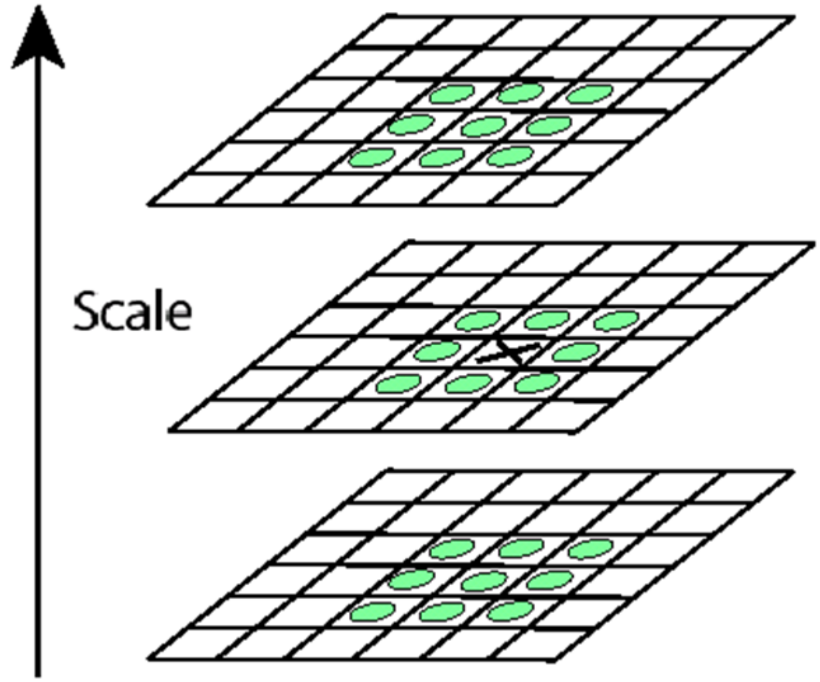
$$G(x, y, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2+y^2}{2\sigma^2}}$$



Note: both kernels are invariant to *scale* and *rotation*

Key point localization

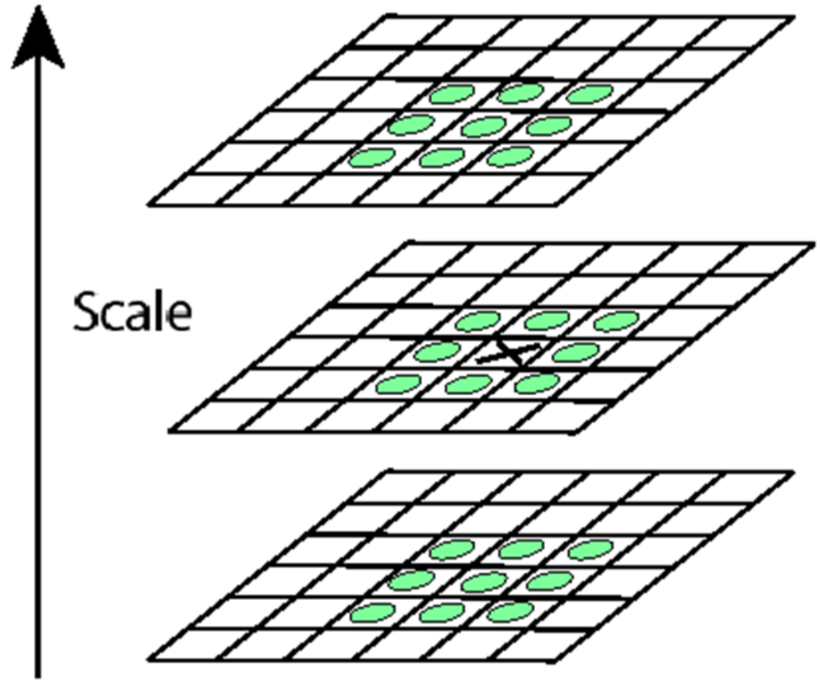
- General idea: find robust extremum (maximum or minimum) both in space and in scale.



Key point localization

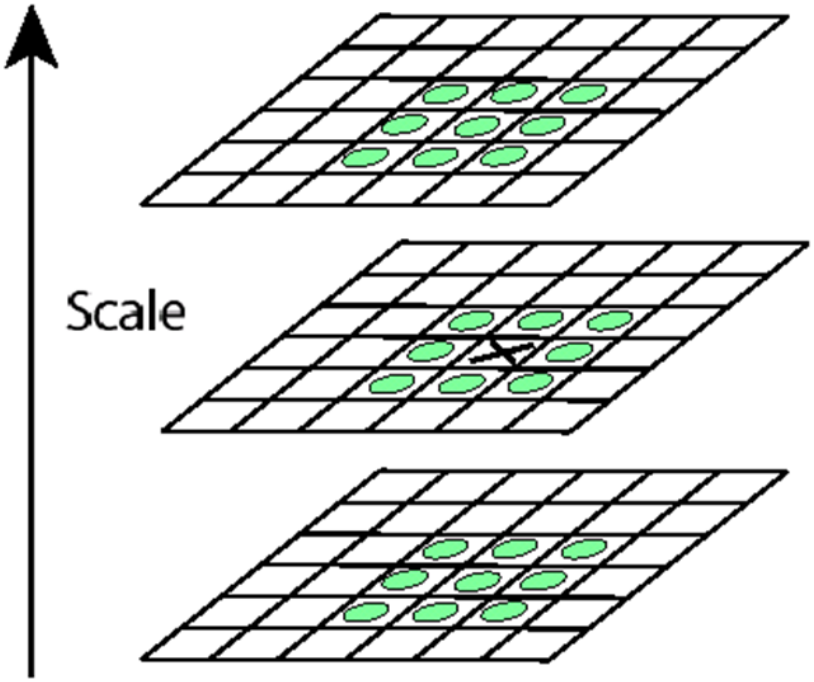
- SIFT: Scale Invariant Feature Transform

- Specific suggestion: use DoG pyramid to find maximum values (remember edge detection?) – then eliminate “edges” and pick only corners.

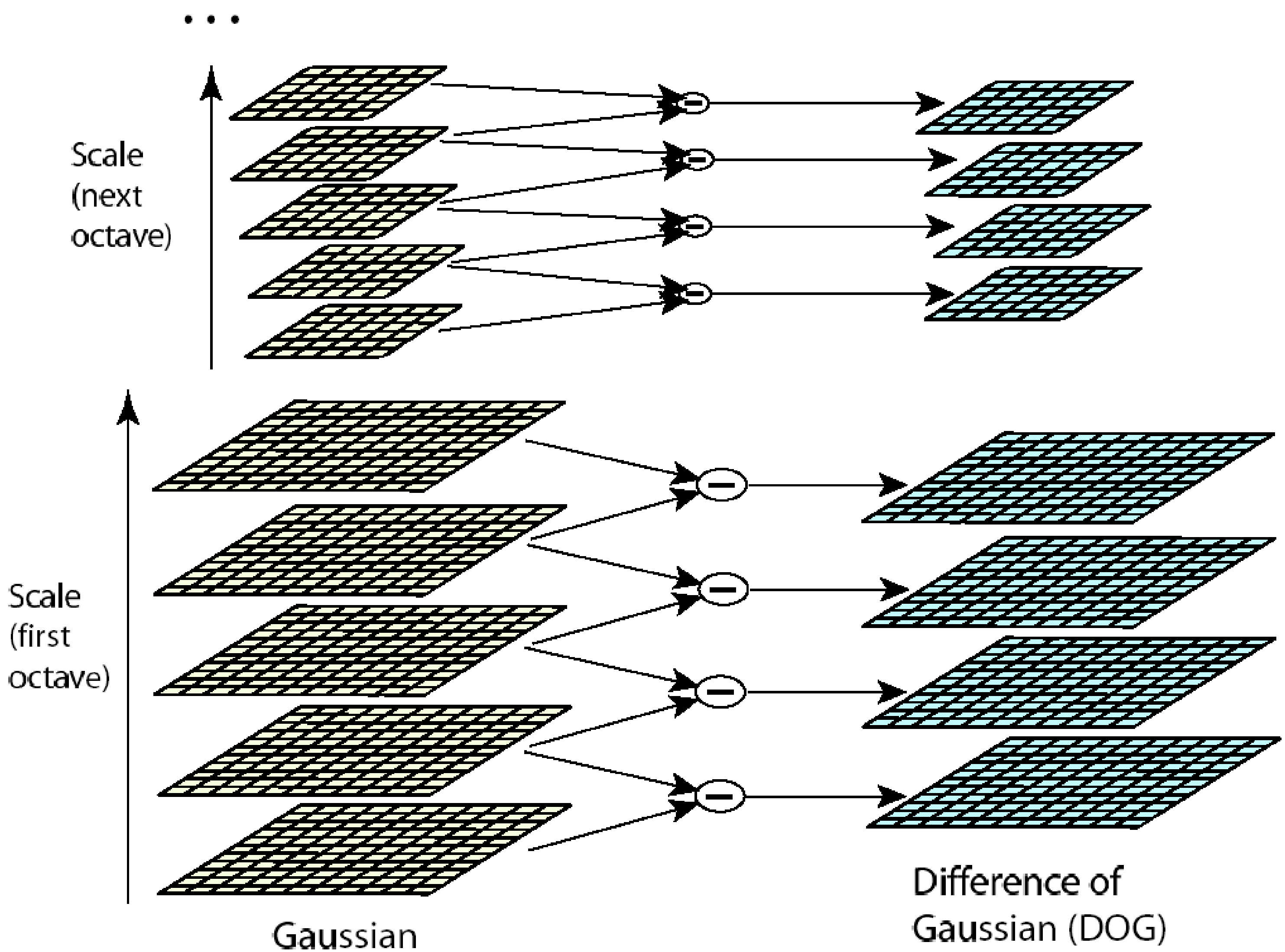


Key point localization

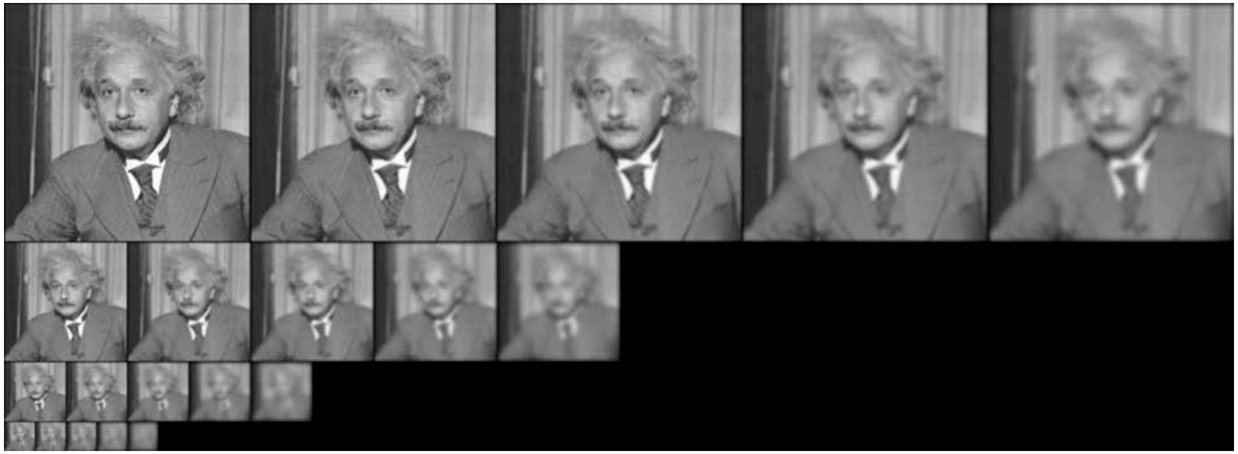
(Each point is compared to its 8 neighbors in the current image and 9 neighbors each in the scales above and below.)



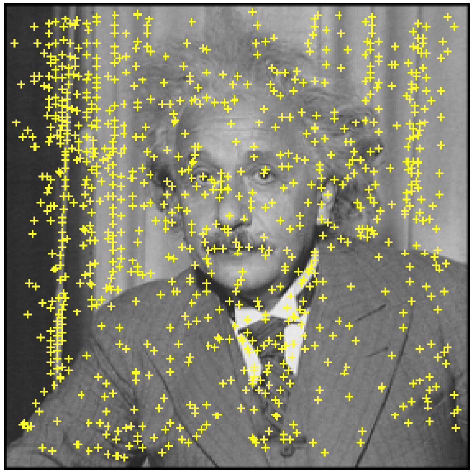
Scale space processed one octave at a time



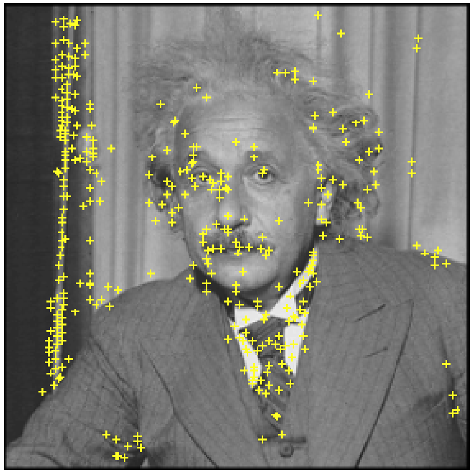
Extrema at different scales



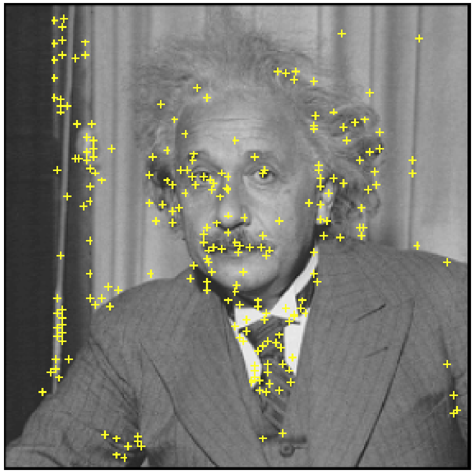
Remove low contrast, edge bound



Extrema points



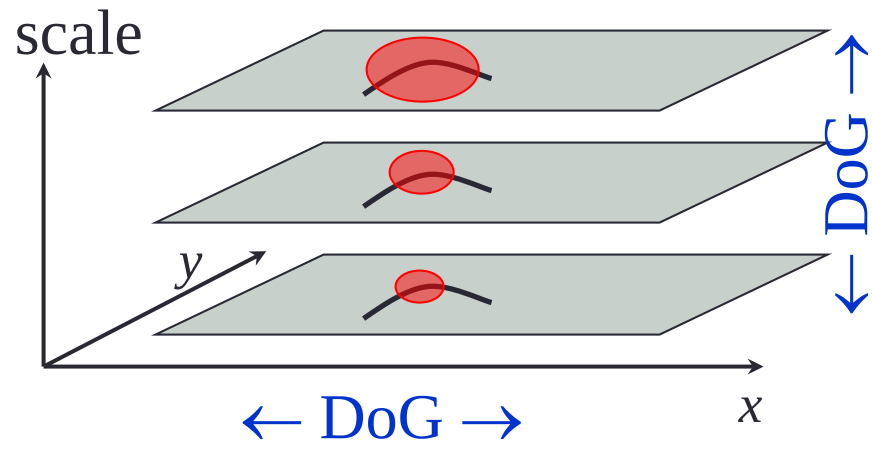
Contrast > C



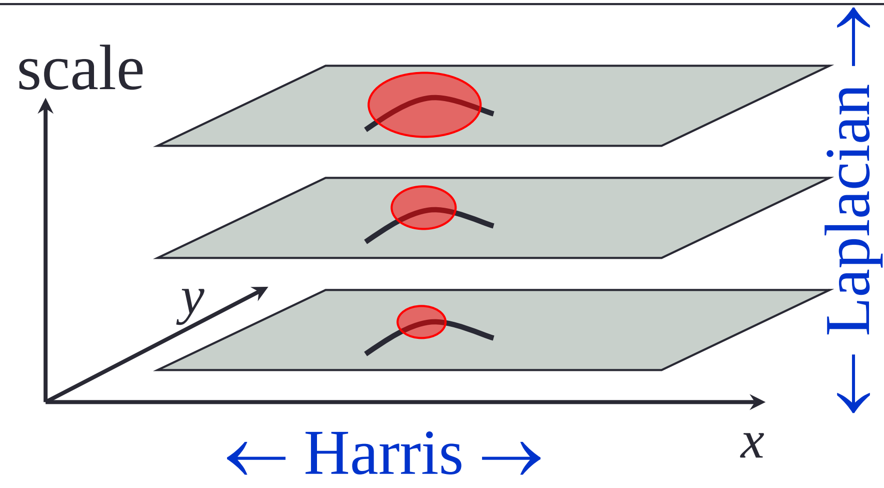
Not on edge

Scale Invariant Detectors

- SIFT (Lowe)[1]
  Find local maximum of:
  – Difference of Gaussians in space and scale



- Harris-Laplacian[2]
  Find local maximum of:
  - Harris corner detector in space (image coordinates)
  - Laplacian in scale



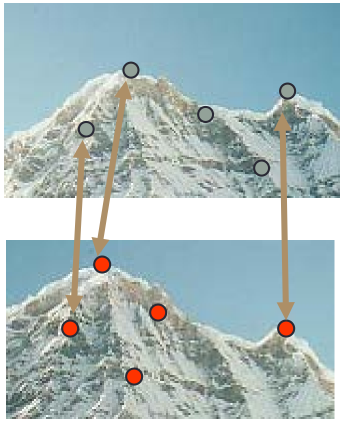
[1] D.Lowe. "Distinctive Image Features from Scale-Invariant Keypoints". IJCV 2004
[2] K.Mikolajczyk, C.Schmid. "Indexing Based on Scale Invariant Interest Points". ICCV 2001

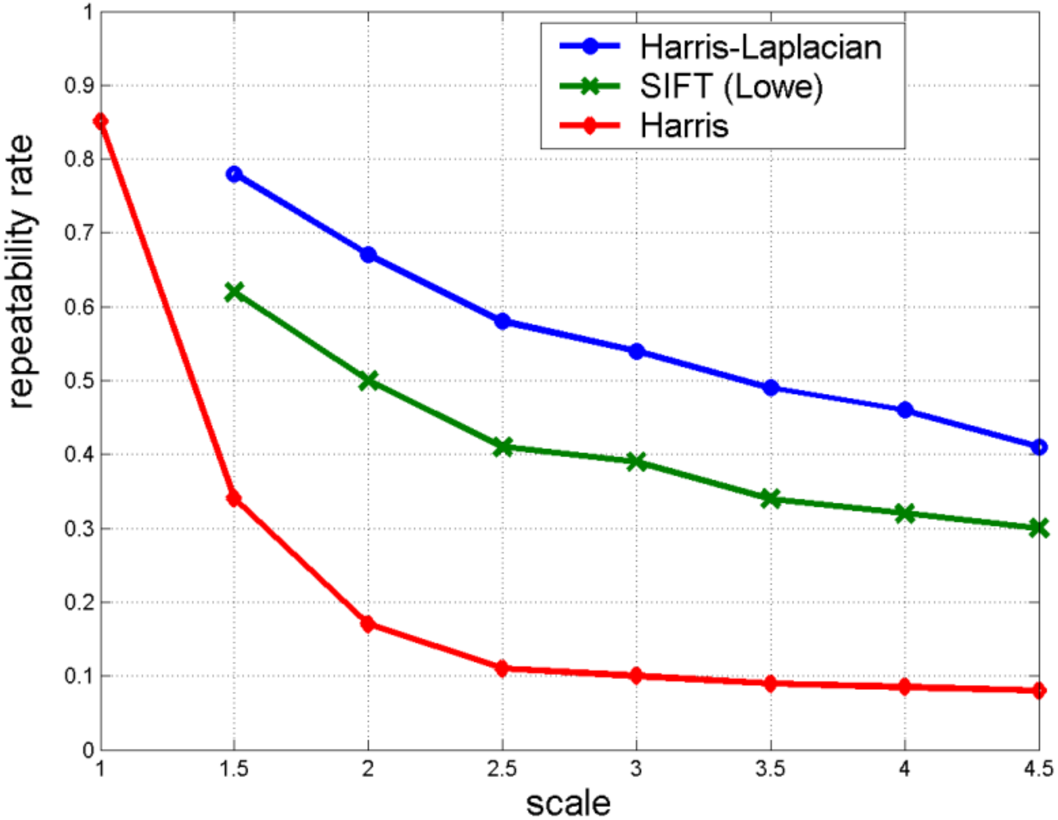
Scale Invariant Detectors

- Experimental evaluation of detectors w.r.t. scale change

Repeatability rate:

$$\frac{\text{\# correspondences}}{\text{\# possible correspondences}}$$





K.Mikolajczyk, C.Schmid. “Indexing Based on Scale Invariant Interest Points”. ICCV 2001

Scale Invariant Detection: Summary

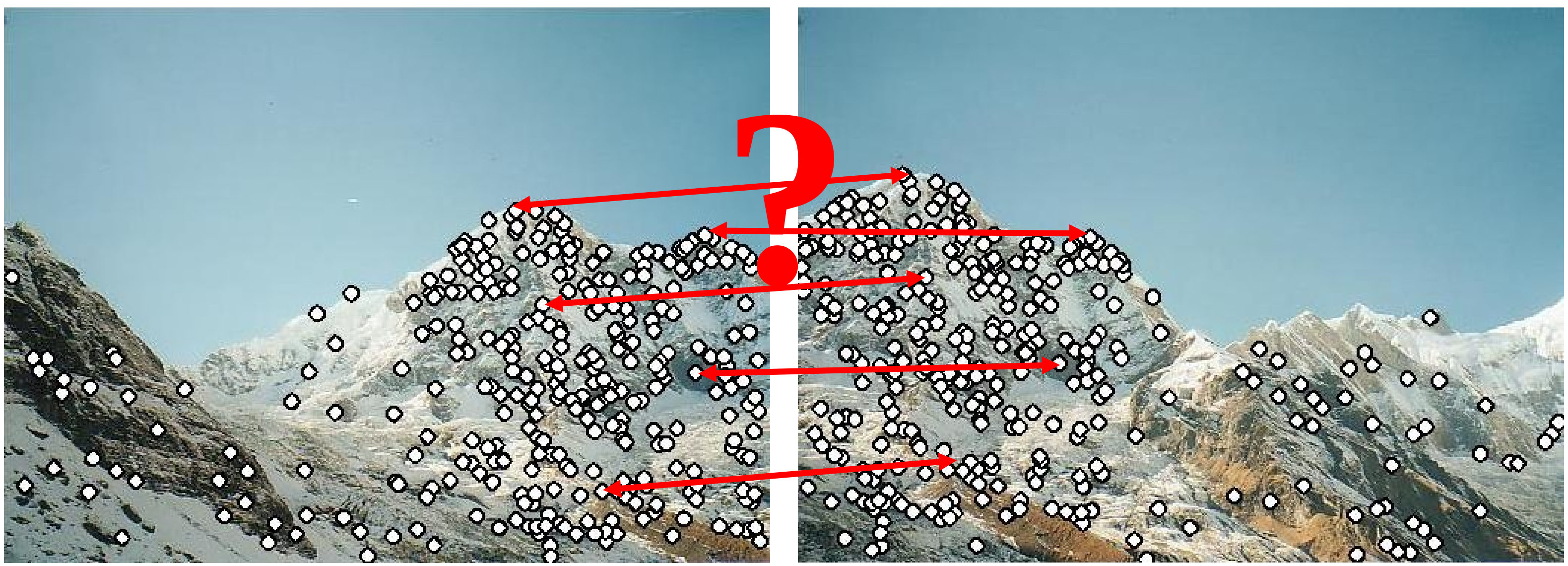
- Given: two images of the same scene with a *large* scale difference between them
- Goal: find the same interest points independently in each image
- Solution: search for maxima of suitable functions in scale and in space (over the image)

Methods:

1. SIFT [Lowe]: maximize Difference of Gaussians over scale and space
2. Harris-Laplacian [Mikolajczyk, Schmid]: maximize Laplacian over scale, Harris' measure of corner response over the image

Point Descriptors

- We know how to detect points
- Next question: How to match them?



Point descriptor should be:
1. Invariant
2. Distinctive

Next time…

- SIFT, SURF, SFOP, oh my…